ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO TÉCNICO DE TELECOMUNICACIÓN, ESPECIALIDAD EN SONIDO E IMAGEN

Título del proyecto:

"DETECCIÓN APROXIMADA DEL LIMBO ESCLEROCORNEAL EN IMÁGENES DE EYE-TRACKING"

Pilar Rufas Gistau

Arantxa Villanueva Larre

Pamplona, 2 de Junio

AGRADECIMIENTOS

A Arantxa, mi tutora, por la oportunidad que me dio al realizar este proyecto y aprender de él, al igual que agradecerle la paciencia que ha tenido a lo largo de estos meses y el tiempo dedicado.

A mis padres, por brindarme siempre la oportunidad de seguir adelante, de hacer lo que me gusta y por tener siempre las palabras adecuadas en los momentos que más lo he necesitado en los últimos años.

A Ainara, Amaia y Raquel por aguantar todo tipo de actitudes y situaciones en estos meses, y sobre todo por estar ahí.

A la gente que he conocido a lo largo de estos cinco años, que han dejado huella por mínima sea, y en especial a Maider y Noemi.

A mis amig@s de siempre, Vicky, Anita, Rubia, Silvia, Iris, Adriana, Ercy, Patry… y a todos los demás que siempre estáis ahí.

A todas aquellas personas que en diversas situaciones me han aguantado durante estos cinco años.

Sobre todo dedicar todo mi tiempo de este proyecto a mi abuela, que gracias a ella también he llegado hasta donde estoy.

Gracias a tod@s!

INTRODUCCIÓN

El presente proyecto tiene como objetivo el cálculo aproximado de la elipse esclerocorneal en imágenes de Eye-tracking.

Las imágenes con las cuales se va a trabajar están obtenidas a partir de un sistema video oculográfico. Como partida común de estas imágenes tendremos en cuenta que se dispondrá de dos reflejos en cada imagen.

A partir de unos datos de entrada, se realizará el procesado de la imagen para poder calcular los puntos que darán la elipse que más se aproxime.

1. EYE-TRACKING

1.1. ¿Qué es Eye-tracking?

Eye-tracking se define como el proceso de medición del movimiento de un ojo en relación con la cabeza. A su vez se puede concluir su descripción como el programa de seguimiento del ojo de un dispositivo para medir la posición de los ojos y sus movimientos.

El concepto general al que nos referimos, se basa en la referencia de tecnologías que permiten monitorizar y registrar la forma en la que una persona mira una determinada escena o imagen.

Al utilizar esta tecnología, el seguimiento se realiza con una cámara e iluminación infrarroja. Un software analiza los movimientos del ojo del usuario.

Estos sistemas tienen un gran potencial de aplicación en varias disciplinas y áreas de estudio, desde el marketing y publicidad hasta la investigación médica o la psicolingüística, así como la interacción hombre máquina

1.2.Dispositivo Utilizado

1.2.1. Historia de los sistemas Eye-tracking

Esta tecnología puede parecer reciente, pero la labor de estudiar el movimiento ocular comenzó un siglo atrás como advierten Jacob y Karn en su estudio [Jacob, 1991]. Últimamente es notorio el creciente interés por la investigación y desarrollo de este tipo de sistemas, probablemente sea debido a la proliferación de soluciones comerciales a precios relativamente asequibles. Hoy en día existen diferentes sistemas en el mercado (Iriscom, Tobii, etc.). Los sistemas de Iriscom, por ejemplo, incluyen el hardware necesario (cámara e iluminadores) y un software específico a precios cercanos a los 8000-11000€. Sin embargo, los avances tecnológicos han hecho posible el desarrollo de

sistemas más baratos. Actualmente se está trabajando en sistemas que permiten el acceso al público mayoritario [D. Beymer y M. Flickner], [R.J.K. Jacob].

Los primeros estudios sobre el movimiento de los ojos se realizaron mediante la observación visual directa (o con ayuda de telescopios o espejos) o introspección. En 1879, Javal [Javal, E.] observó los movimientos del ojo al leer, manteniendo un ojo cerrado y junto a este un micrófono Delabarre [Delabarre, E. B.] también estudió estos movimientos sellándolos con un quimógrafo. Más adelante, en 1900, Huey [Huey, E. B.], construyó un sistema de Eye-tracking utilizando una especie de lente de contacto con un agujero para el usuario. La lente estaba conectada a un puntero de aluminio que se movía en respuesta al movimiento del ojo.

Delabarre y Huey fueron capaces de obtener una primera visión muy valiosa de la función y características de los movimientos oculares, pero sus métodos fueron criticados por resultar demasiado invasivos. Para superar estos inconvenientes Dodge y Cline [Dodge, R., y Cline, T. S.] en 1901 registraron el movimiento de los ojos de manera más precisa y no invasiva. Dodge [Dodge, R.] emitía líneas de luz a los ojos y grababa su reflejo mediante la fotografía (gran aportación hasta los siguientes avances en los 70 con la era digital). Esta fue la primera investigación del movimiento ocular utilizando la reflexión de la córnea, método que hoy en día se sigue utilizando.

Gracias a la evolución de la década de 20 en laboratorios Chicago y Standfor se iniciaron los primeros registros de movimientos oculares en tres dimensiones capturando dos imágenes del ojo simultáneamente. Más adelante, se consiguió descomponer la reflexión de los haces de luz de un ojo en sus componentes horizontal y vertical. Buswell [Buswell, G. T.] en 1935 utilizó esta descomposición de la reflexión del haz para medir las rutas de análisis de un sujeto al observar una imagen.

En la década de los 60 se realizan investigaciones utilizando otra vez técnicas invasivas para mejorar la exactitud de las técnicas anteriores. En este caso, lentes de contacto junto con espejos o incluso rollos de alambre, que utilizaron investigadores como Yarbus [Yarbus, A. L.] o Fender [Fender, D. H.].

En la década de 1970, la investigación de seguimiento del ojo se expandió rápidamente, en particular la investigación del movimiento del ojo en la lectura (Rayner [Rayner, K.]). Esta expansión se produjo también por las nuevas técnicas que empezaron a utilizarse, como el escaneo del ojo mediante el uso de una cámara. Con esta técnica se conseguía alto contraste entre el iris y la esclerótica facilitando su separación.

De esta década destacan investigadores como Taylor S. E. [Taylor, S. E.] quien inventó un dispositivo mecánico para traducir los movimientos del ojo en permanentes o Merchant y otros [J. Merchant *et al.*] quienes a través de su oculómetro calculaban la posición del centro de la pupila y los reflejos en la imagen.

Todos los métodos anteriores miden el movimiento de los ojos respecto a la cabeza, por lo que los investigadores debían de utilizar sistemas para permanecer con la cabeza inmóvil. Esto fue obviado en esta década al utilizar dos características ópticas del ojo en movimiento, ya que los reflejos de los haces de luz permanecen invariantes ante traslaciones de la cabeza pero varían con rotaciones del ojo (según el estudio de Merchant).

Actualmente la tecnología de Eye-tracking principalmente sigue siendo reservada a tareas de investigación y de desarrollo de productos, aunque se espera que su evolución sea muy rápida en los próximos años, hasta el punto de poder llegar a convertirse en interfaz de uso cotidiano en dispositivos de consumo. Según los expertos, hoy en día nos encontramos en la cuarta era de los sistemas de Eye-tracking caracterizada por el gran número de nuevas aplicaciones de las cuales las más representativas serán expuestas a continuación.

Atendiendo al criterio de Andrew T. Duchowsky [Andrew T Duchowsky] se pueden distinguir dos vertientes de aplicación de sistemas de Eye-tracking: diagnósticas (pasivas, de análisis y diagnóstico) e interactivas (como medio de interacción con el computador).

En el primero de los casos, el sistema de Eye-tracking proporciona la evidencia objetiva y cuantitativa de los procesos relacionados con la visión y la percepción, midiendo el movimiento de los ojos y la atención al realizar otra tarea independiente. Es decir, funciona como herramienta de evaluación: la información extraída ya no requiere de un análisis y respuesta en tiempo real por el sistema sino que es registrada para su posterior análisis e interpretación. Esta técnica es aplicada por ejemplo en campos de marketing o para diseños de páginas web, mediante mapas de calor que miden las zonas que llaman más la atención al observar una escena determinada (fig.1).

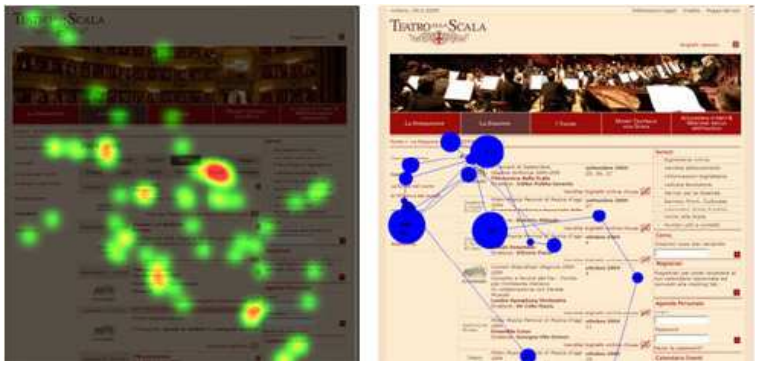


Fig. 1 *Densidad de fijaciones en una página web*

Cuando se utiliza como medio interactivo sirve como un dispositivo de entrada sustituyendo al ratón. Aunque la precisión del sistema de Eye-tracking como dispositivo de entrada dista de la de otros, como el ratón, puede tener numerosas aplicaciones prácticas, tales como su uso en entornos de realidad virtual o por usuarios con discapacidad motriz (como la esclerosis múltiple). Pero además, incluso en determinadas operaciones como la selección de objetos de la interfaz, la mirada puede resultar más rápida que la acción mediante el ratón.

1.2.2. Clasificación de las técnicas de Eye-tracking

Los sistemas de Eye-tracking se pueden dividir en tres categorías por las metodologías de medición de movimientos oculares que impliquen: electro-oculografía (EOG), lentes de contacto/ bobinas de búsqueda y foto-oculografía (FOG) o video-oculografía (VOG).

Una de las características de los sistemas Eye-tracking requerida para la comodidad del usuario, como hemos visto en las diferentes técnicas utilizadas durante la historia, es el carácter no invasivo de los sistemas. De esta forma podríamos agrupar las tres categorías de las metodologías antes mencionadas en dos: sistemas invasivos y no invasivos.

a) Sistemas invasivos:

Entre los sistemas invasivos que requieren el contacto físico con el sujeto encontramos los sistemas basados en lentes de contacto/bobinas de búsqueda y sistemas basados en electro-oculografía (EOG). Generalmente estos sistemas realizan mediciones de mayor precisión.

a.1) Sistemas basados en lentes de contacto:

Esta es una de las técnicas más precisas que implica el contacto físico con el ojo a través de un mecanismo basado en lentes de contacto, pero inevitablemente estos sistemas resultan muy incómodos para los usuarios.

Esta técnica implica asignar un objeto de referencia mecánico u óptico para ser montado en la lente de contacto que luego se coloca directamente en el ojo, como por ejemplo fósforos que reflejan, diagramas de línea o bobinas inductoras. El método principal emplea bobinas de búsqueda y el movimiento se mide a través de un campo electromagnético. A pesar de que es el método más preciso (con una precisión de 5-10 arco segundos en un rango limitado de 5°) es el método más invasivo.

Estas técnicas no parecen ofrecer una buena solución a la integración a la tecnología de Eye-tracking. A su vez, los problemas de salud derivados de los campos magnéticos de alta frecuencia no han sido resueltos todavía existiendo ciertas dudas sobre su efecto en el organismo humano. Muchos expertos expresan sus reservas sobre esta tecnología aún y cuando se lograsen solucionar estos problemas por tratarse de una técnica excepcionalmente invasiva.

a.2) Electro-oculografía (EOG)

Esta técnica fue una de las más utilizadas hace unos cuarenta años, consiste en la medida de diferencias de potencial eléctricas en la piel detectadas por electrodos colocados alrededor del ojo. Esta técnica en principio ofrecería gran libertad de movimientos, pero al necesitar detectar la posición de la cabeza también se pierde esta propiedad. Estas diferencias de potencial se producen con movimientos de los ojos pero no son constantes y sus variaciones hacen difícil de usar EOG para medir los movimientos oculares lentos y la detección de dirección de la mirada. Sin embargo, es

una técnica muy sólida para medir los movimientos oculares sacádicos asociados con cambios de la mirada y la detección de parpadeos. Además, esta técnica requiere una potencia de cálculo muy baja y se puede utilizar en ausencia de iluminación.

b) Sistemas no-invasivos:

Entre los sistemas no invasivos encontramos aquellos basados en técnicas de video o foto oculografía. El carácter no invasivo de estos sistemas hace que hoy en día sean los más utilizados. En este apartado serán introducidos los sistemas de video-oculografía, no obstante al tratarse de la tecnología que utilizaremos para capturar las imágenes utilizadas en este proyecto, en el apartado siguiente serán desarrollados en detalle.

b.1) Foto oculografía (POG) y video-oculografía (VOG)

Esta técnica incorpora una cámara o dispositivo de adquisición de imágenes para tratar de determinar el movimiento de los ojos utilizando las imágenes obtenidas por dicho dispositivo. Para ello, se analiza el comportamiento de características del ojo humano como el centro de la pupila, el contorno del iris o las reflexiones de la córnea (fig.2) e intentando establecer una relación entre estas y el punto donde se fija la mirada (*PoR*).

Estos sistemas están basados en la captura de imágenes por lo que requieren una iluminación del usuario adecuada. Algunos de estos sistemas cuentan con una o varias fuentes de iluminación aunque también existen sistemas que funcionan solamente con la luz visible.

En el caso de utilizar iluminadores, generalmente la luz emitida por estos es infrarroja. Esta se refleja en los ojos y es capturada por la cámara de vídeo. La información capturada se analiza para extraer la rotación de los ojos en función de los cambios de posición de las reflexiones.

Distinguimos diferentes Eye-trackers por la característica del ojo que analizan. Aprovechando la reflexión del ojo, hay Eye-trackers que emplean el primer reflejo de la córnea (primera imagen de Purkinje). Esta es la reflexión más fácil de detectar pero, también se producirán más reflejos de los cuales el primero y cuarto serán los más relevantes.

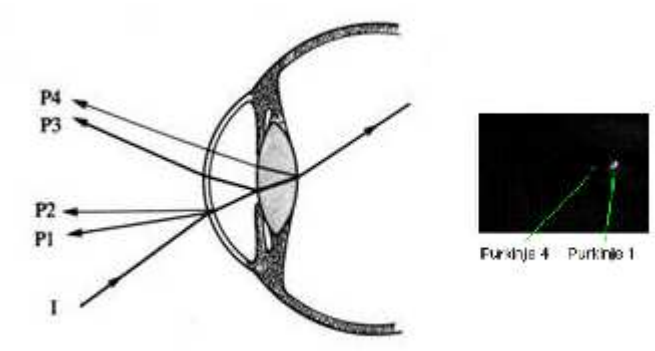


Fig.2 *Reflejos de Purkinje.*

No obstante para capturar las imágenes utilizadas en este proyecto utilizaremos uno de los métodos oculográficos que mejor funcionan el cual combina este primer reflejo de la córnea (1ª imagen de Purkinje) y la información del centro de la pupila.

Así mismo, las técnicas de Eye-tracking mediante video-oculografía podrían dividirse en función de su objetivo:

- Técnicas que miden los movimientos del ojo
- Técnicas que determinan el punto exacto donde se fija la mirada, o cálculo del *PoR (Point of Regard)*.

La primera técnica determina la rotación del ojo, esto es, la orientación del ojo respecto a la cabeza. En la segunda, se calcula el punto a donde el sujeto mira en el espacio; para tal fin será necesario el seguimiento independiente de la posición de la cabeza (en 3D) además de conocer la rotación del ojo.

Entre los sistemas basados en video-oculografía distinguimos los basados en modelos matemáticos y los basados en modelos polinómicos. Cualquiera de estos sistemas podemos diseñarlo para ir montados en la cabeza, como un casco, esta configuración consta de una cámara para el seguimiento del movimiento del ojo y otra cámara capturando la escena a la que el usuario mira, o como sistema remoto aunque en muchos casos habría que mantener la cabeza quieta utilizando un reposa cabezas debido a la falta de robustez de estos sistemas ante desplazamientos de la cabeza.

Ambas soluciones conllevan que el sistema pierda el carácter no invasivo, es decir, perdería la cualidad que hace que estos sistemas sean los más utilizados.

1.3.Sistemas video oculográficos (VOG)

Los elementos básicos de un sistema de Eye-tracking basados en video-oculografía son, como ya hemos comentado anteriormente, las fuentes de iluminación y el sistema de adquisición de imágenes empleado. En los últimos años han sido desarrollados diversos algoritmos utilizando sistemas video oculográficos. Aunque todos ellos tengan la misma base admiten múltiples variaciones como el tipo de iluminación, el número de iluminadores o cámaras utilizadas, la localización de cada uno de los elementos, etc.

Hay dos tipos de iluminación de uso común utilizados para el seguimiento de la mirada: la luz visible y la infrarroja. La utilización de imágenes del espectro visible es una actitud pasiva la cual captura la luz ambiental reflejada en el ojo. La mejor característica para realizar un seguimiento con este tipo de iluminación es el contorno entre el iris y la esclerótica conocido como el limbo (fig. 3 izda.).

Las tres características más relevantes del ojo son la pupila, el iris, y la esclerótica. La utilización de este tipo de iluminación se complica por el hecho de que la luz ambiente puede contener varios componentes de luz especulares. En cambio, al utilizar iluminación infrarroja se elimina la reflexión especular no controlada, y permite que la iluminación de los ojos sea uniforme, no siendo perceptible para el usuario. Un beneficio adicional de la proyección infrarroja es que la característica más fuerte, en cuanto a los contornos de la imagen, es la pupila y no el limbo (fig. 3 dcha.); la

esclerótica y el iris reflejan la luz infrarroja, en cambio, la luz visible solamente la esclerótica. El seguimiento del contorno de la pupila es preferible dado que el contorno de la pupila es más pequeño y más definido que el del limbo. Por otra parte, debido a su tamaño, la pupila es menos probable de ser ocluida por los párpados. La principal desventaja de técnicas de imagen infrarroja es que no se puede utilizar al aire libre, durante en el exterior, debido a la iluminación infrarroja ambiente.

Fig.3. *Detección del contorno exterior del iris (izda.) y de la pupila (dcha.)*

El seguimiento de la mirada utilizando iluminación infrarroja permite dos técnicas para la detección de la pupila: la técnica de pupila brillante y la técnica de pupila oscura (fig. 4). También puede utilizarse la combinación de las dos técnicas para conseguir un sistema más preciso [C. Morimoto *et al.]*.

La técnica de pupila brillante ilumina el ojo con una fuente IR que se encuentre muy cerca del eje de la cámara. El resultado de la iluminación es tal que la pupila está claramente delimitada como una región brillante debido a la naturaleza de la parte reflectiva, la parte posterior, del ojo. La técnica de pupila oscura por el contrario ilumina el ojo con una fuente situada fuera del eje de la cámara siendo así la pupila la parte más oscura del ojo en la imagen. En ambos métodos, la reflexión especular de la primera superficie de la fuente de iluminación fuera de la córnea (el elemento exterior-óptico del ojo) también es visible. El vector entre el centro de la pupila y del reflejo de la córnea se suele utilizar en lugar del centro de la pupila como única indicación, para detectar la dirección de la mirada.

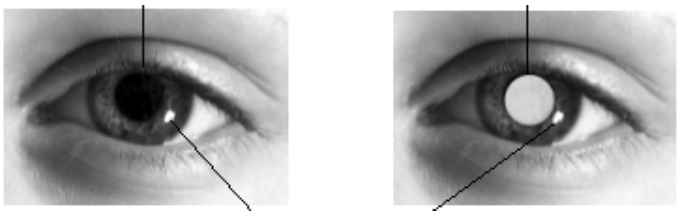


Fig.4 *Técnica de pupila oscura (izda.) y técnica de pupila brillante (dcha.)*

La diferencia de posición entre el centro de la pupilla y el reflejo producido por la córnea (primera reflexión de Purkinje) varía con la rotación del ojo pero con un pequeño movimiento de la cabeza puede mantenerse prácticamente constante. Por ello, una de las soluciones vigentes es las imágenes dual-Purkinje en las cuales se miden la primera y cuarta reflexión de Purkinje para separar las traslaciones y rotaciones de los movimientos del ojo. Sin embargo, esta solución sigue sin ser lo suficientemente precisa con lo que el usuario deberá utilizar indistintamente un reposa cabezas.

3.1.1 Configuraciones posibles

Todas estas técnicas anteriormente citadas son aplicables a cualquiera de las configuraciones de sistemas que se van a exponer a continuación.

En cuestiones de diseño, los sistemas VOG podemos agruparlos en dos: aquellos sistemas de Eye-tracking montados en la cabeza (fig.5) y los sistemas remotos, que serán detallados en el apartado 1.3.2. Los primeros son preferidos para aplicaciones que requieren grandes y rápidos movimientos de la cabeza; no es conveniente utilizarlos en aplicaciones que requieren una continua mirada de vigilancia durante largos períodos de tiempo (por ejemplo, el control del cursor para las personas con discapacidad motriz) o en aplicaciones que involucran a los niños. Para estas aplicaciones, los sistemas a distancia son los preferidos.

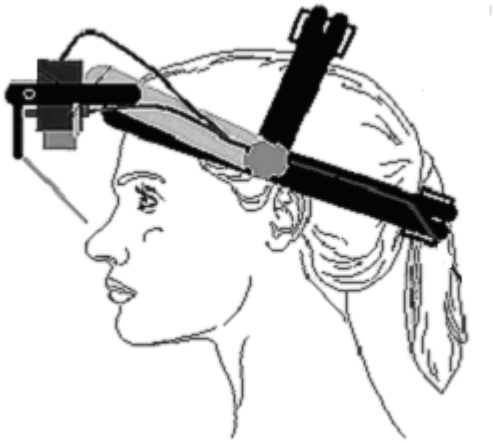


Fig. 5 *Sistema de Eye-tracking montado en la cabeza.*

Recientemente, la invasión del seguimiento de la mirada ha sido reducida considerablemente con los avances en la miniaturización de los sistemas que son montados sobre la cabeza en los eye trackers. Los sistemas remotos reducen la invasión respecto a los anteriores, sin embargo, pueden sufrir de una reducción de la precisión. Teniendo en cuenta estos avances, el obstáculo restante más importante es el costo. Los precios de estos sistemas no están puramente asociados con el hardware, como en el precio de una cámara digital de alta resolución, sino con un software personalizado, integrados a veces con especializados procesadores digitales para obtener rendimiento de alta velocidad.

Entre los sistemas remotos de Eye-tracking, la configuración más simple consta de una sola cámara y una fuente de luz (la misma configuración que la figura 6a con un LED (light emitting diode) menos). No obstante, para calcular la dirección de la mirada con alta precisión no es suficiente el uso exclusivo de este sistema. Para satisfacer esta limitación habrá que mantener la cabeza fija en relación con el sistema y habrá que estimar la distancia entre el ojo y la cámara. Por lo general estos sistemas suelen calcular la dirección de la mirada utilizando el vector del centro de la pupila al centro de

la reflexión de la córnea con respecto al eje de la cámara o bien disponer de medios para estimar la posición del ojo en el espacio con ayuda de espejos por ejemplo.

La segunda configuración (fig.6a) que puede presentar un sistema remoto es la configuración más simple que permite la estimación del *PoR* a partir de los centros de la pupila y reflejos que no requiere de un cálculo adicional. Se trata de la configuración de partida que será analizada en este proyecto, que cuenta con una cámara y dos iluminadores una a cada lado de la pantalla. Aunque esta configuración no implique ninguna restricción en cuanto a movimientos de la cabeza y no requiera la utilización de ningún dispositivo adicional para estimar la posición del ojo en el espacio, para conseguir una precisión del sistema óptima, una de las dos soluciones será necesaria.

El resto de los sistemas que se van a presentar cuentan con más de una cámara para estimar la posición del ojo en 3D. El primero cuenta con dos cámaras estéreo (fig.6c). Esta configuración tiene dos algoritmos para acomodar los cambios de posición de los ojos:

1.- Basándonos en la posición del globo ocular obtenido, la dirección de la mirada calculada puede ser actualizada ante cambios de posición de la cabeza por una propuesta dinámica del modelo computacional de posición de la cabeza de compensación.

2.- A partir del seguimiento de la mirada en 3D propuesto, podemos estimar los ejes virtuales en 3D directamente sin funciones de asignación.

Ambos métodos propuestos pueden alcanzar la precisión de menos de 1 grado a una distancia de unos 60 cm a la pantalla.

El sistema contiene los iluminadores en el eje de la cámara por lo que para la detección de la pupila se utilizará la técnica de pupila brillante.

El siguiente sistema cuenta con varias cámaras y dos iluminadores y se denomina sistema multi-cámara (fig.6b), cada una de estas puede estimar la mirada de forma independiente, por lo tanto, permite grandes movimientos de la cabeza. La precisión de este sistema es también de menos de 1 grado, permaneciendo a una distancia de unos 60cm del sistema. Esta configuración contiene los iluminadores fuera del eje de la cámara por lo que para la detección de la pupila se utilizará la técnica de pupila oscura.

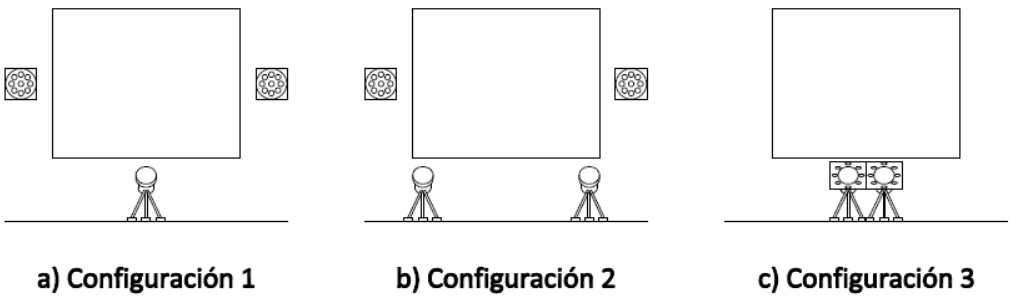


Fig.6 *Esquema de tres de las posibles configuraciones*

3.2 Descripción del sistema de Eye-tracking utilizado

El sistema Eye-tracker (fig.7) que se ha utilizado para la realización de este proyecto, como se ha comentado en el apartado 1.3, es un sistema basado en video-oculografía. Concretamente esta formado por dos fuentes de iluminación, LEDs infrarrojos, posicionados en los laterales de la pantalla a mitad de altura de la misma y una cámara de vídeo delante de la pantalla en el centro orientado hacia el ojo del usuario.

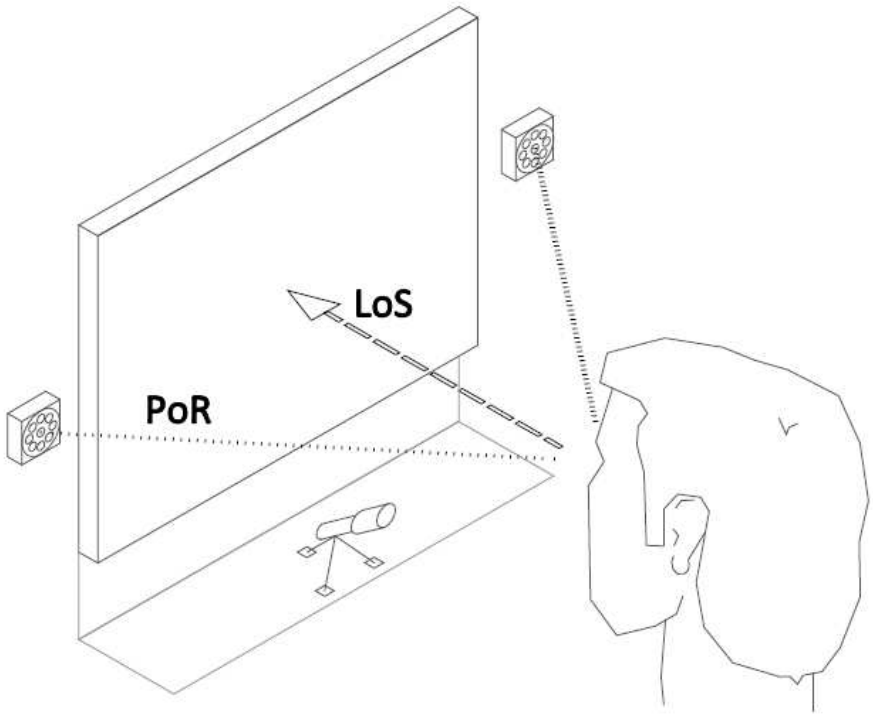


Fig. 7 *Configuración del sistema de partida utilizado*

La luz infrarroja emitida por los LEDs es imperceptible por el ojo humano evitando distracciones del usuario, pero su espectro está centrado en 890nm, es decir, dentro del rango de actuación de la cámara. Además, esta luz no es nociva para los usuarios.

El primer paso del análisis será la obtención de la imagen (fig.8) a través del sistema de video-oculografía. Utilizando algoritmos de procesado de imagen se extraen de las imágenes las características del ojo que se asocien al movimiento del mismo. En este caso calculamos la posición del centro de la pupila y del primer reflejo de la córnea producido por cada uno de los LEDs (primera imagen de Purkinje).

Fig. 8 *Ejemplo de imagen capturada por el sistema*

3.2.1 Calibración

Para iniciar el seguimiento de la mirada es necesaria antes una sesión de calibración debido a que los coeficientes de la ecuación del cálculo del *PoR* se verán modificados en cada sesión dependiendo de la posición de la cámara, del usuario y de la pantalla, y los LEDs.

La ecuación de calibración viene en forma de coeficientes indeterminados, como una expresión polinómica cuyas variables son combinaciones lineales de las características de la imagen, el centro de la pupila y de los reflejos, multiplicados por unos coeficientes a calcular.

$$\mathbf{P_0 = (C_0)^T\ Fe}$$

P_0: *PoR*, punto donde miras
C_0: vector de coeficientes desconocidos a calcular en la calibración
Fe: vector de características de la imagen y posibles combinaciones lineales de las mismas.

La sesión de calibración (fig.9) consiste en que el usuario observe sucesivamente una serie de puntos (una cuadrícula generalmente) calculando las posiciones del centro de la pupila y los reflejos al mirar cada una de las fijaciones, conocidas por el sistema.

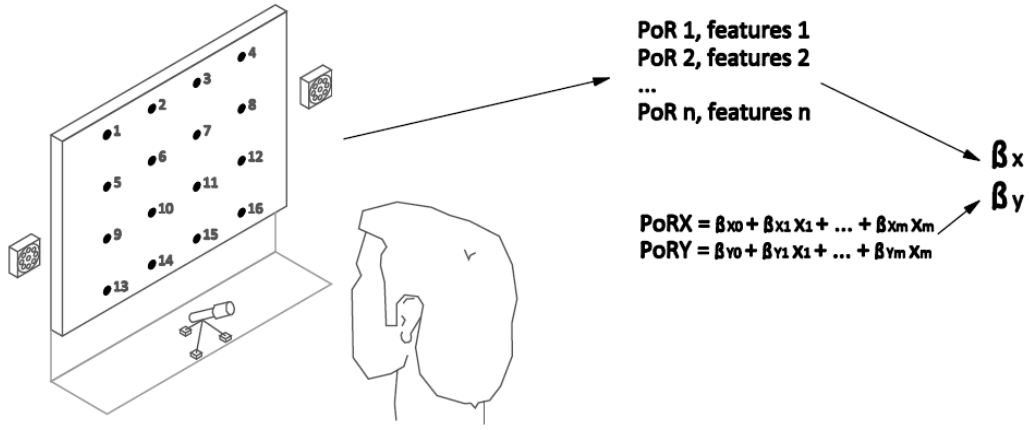


Fig. 9 *Esquema del grid de calibración*

2. SHAKTI

El sistema Shakti es un software desarrollado por la Universidad Pública de Navarra que realiza el seguimiento de la mirada permitiendo el uso de ordenador mediante el ojo, moviendo el cursor con la mirada a tiempo real. Shakti ha sido desarrollado para facilitar la comunicación entre personas discapacitadas permitiéndoles el control total del ordenador mediante el movimiento del ojo utilizando el sistema VOG antes descrito para colocar el puntero del ratón en la pantalla.

Una vez el usuario se encuentre en el lugar de la pantalla deseado, puede realizar “click” del ratón, bien parpadeando o bien por latencia.

El hadware que utilizaremos, como ya hemos citado anteriormente, cuenta con una cámara (PointGrey) y dos iluminadores IR a ambos lados de la pantalla. Tanto la cámara como dichos iluminadores serán conectados a una fuente de alimentación. La cámara se sitúa sobre una base delante de la pantalla y se conecta al puerto Firewire del ordenador. El objetivo que inicialmente se utiliza es de 35mm.

Existen tres modos de funcionamiento de la aplicación PJ1: modo Usuario, Cuidador y Mantenimiento. En el primer modo el usuario puede habilitar/deshabilitar el control del cursor del ojo y realizar las calibraciones pertinentes. El modo Cuidador permite configurar la aplicación adaptándola a cada usuario.

El monitor de la pantalla utilizado es de 17’’ (337.9H x 270.33V mm) con una resolución de 1280 x 1024 píxeles. Es decir, 0.26398 mm/px.

La cámara utilizada con sensor de 1/3 (3.6 x 4.8 mm) tiene una resolución de la imagen de 1024 x 768 píxeles. Por lo que las dimensión del píxel de la imagen serán de 7.4 μm.

3. DESARROLLO Y ETAPAS

3.1.OBJETIVO

Este proyecto pretende calcular la elipse aproximada del limbo esclerocorneal en imágenes de Eye-tracking.

Se partirá en cualquier caso de los mismos requisitos. Estos son tanto tener la imagen capturada por el sistema descrito en apartados anteriores, como los puntos correspondientes al centro de la pupila y a los dos reflejos proporcionados por los iluminadores infrarrojos del sistema.

Para el análisis con el que se va a proceder, se plantea como prioridad realizar la segmentación de la imagen, para poder obtener los mejores resultados de los bordes posibles, y de esta manera la posterior búsqueda de los puntos pertenecientes al limbo será más sencilla.

Una vez que se tienen dichos puntos se pretende que la aproximación a la elipse sea sencilla.

Queda claro que para conseguir el resultado óptimo la búsqueda y elección de puntos debe de seguir el algoritmo que mejor resultados obtenga *a posteriori*.

3.2.PLANTEAMIENTO INICIAL Y DESARROLLO

3.2.1. Programa Principal

Finalmente para poder confirmar que el algoritmo es robusto y fiable, se realizarán una serie de comparaciones respecto a parámetros del programa y sobre la elipse calculada mediante otros métodos.

Para ello inicialmente se va a explicar el algoritmo planteado que sigue la imagen a lo largo de su proceso hasta llegar a la obtención de la elipse.

Como ya se ha comentado en apartados anteriores como entrada del programa se dispone de la imagen, el centro de la pupila y sus reflejos, por lo que la llamada al programa principal se realizará mediante la llamada a la función:

[a,b,xc,yc,phi]=imágenes (img,x,y,x1,y1,x2,y2)

```
%Parámetros de entrada:
%  img: imagen de entrada.
% x e y: Coordenadas que pertenecen al centro de la pupila.
% x1 e y1: Coordenadas que pertenecen al centro de un reflejo
% x2 e y2: Coordenadas que pertenecen al centro de otro de los reflejos.

% Parámetros de salida:
% a: semieje mayor de la elipse.
% b: semieje menor de la elipse.
```

% xc e yc: coordenadas centro de la elipse
% phi: ángulo de la elipse.

Dicho algoritmo se basa en calcular a partir de los datos de entrada una serie de parámetros que más tarde serán utilizados para la elección del procesado de la imagen.

Si la distancia (r) entre los reflejos es muy pequeña se realizará previamente una apertura a la imagen, si no se trabajará con la imagen original. La finalidad de realizar la apertura se debe a que cuando los reflejos están muy próximos al centro de la pupila es muy difícil evitarlos, y por tanto la cantidad de ángulos para trabajar posteriormente sería muy pequeña.

Tras este paso se realiza un filtrado Canny de la imagen para obtener los bordes.
A partir de dicha imagen se obtiene una nueva imagen que es con la que se sigue trabajando para la obtención de los posibles puntos candidatos de la elipse.

Una vez que se tienen almacenados los puntos se realizan tres tipos de restricciones para la eliminación de puntos.

La primera se basa en eliminar aquellos puntos en los cuales la componente horizontal del gradiente es nula, ya que en ese caso el punto no es valido.

La segunda restricción se realiza a partir de la media y la varianza de la distancia de todos los puntos respecto al centro de la pupila. Si la distancia de cada punto al centro se encuentra dentro de un umbral el punto es válido, si no se desecha.

El tercer método que se ha comprobado no corresponde a ninguna restricción, si no al cálculo de la elipse con todos los puntos candidatos.

Todos los puntos que han quedado tras este proceso son los que se utilizarán para obtener los parámetros importantes para poder formar la elipse. Por lo que los semiejes, el centro y el ángulo de inclinación de la elipse vendrán definidos por dichos puntos.

Para finalizar se dibujará la imagen original con la aproximación de la elipse sobre ella.

Antes de continuar explicando los siguientes scripts que son llamados a partir del principal se muestra un diagrama de flujo de la función principal que sigue el programa (fig.10).

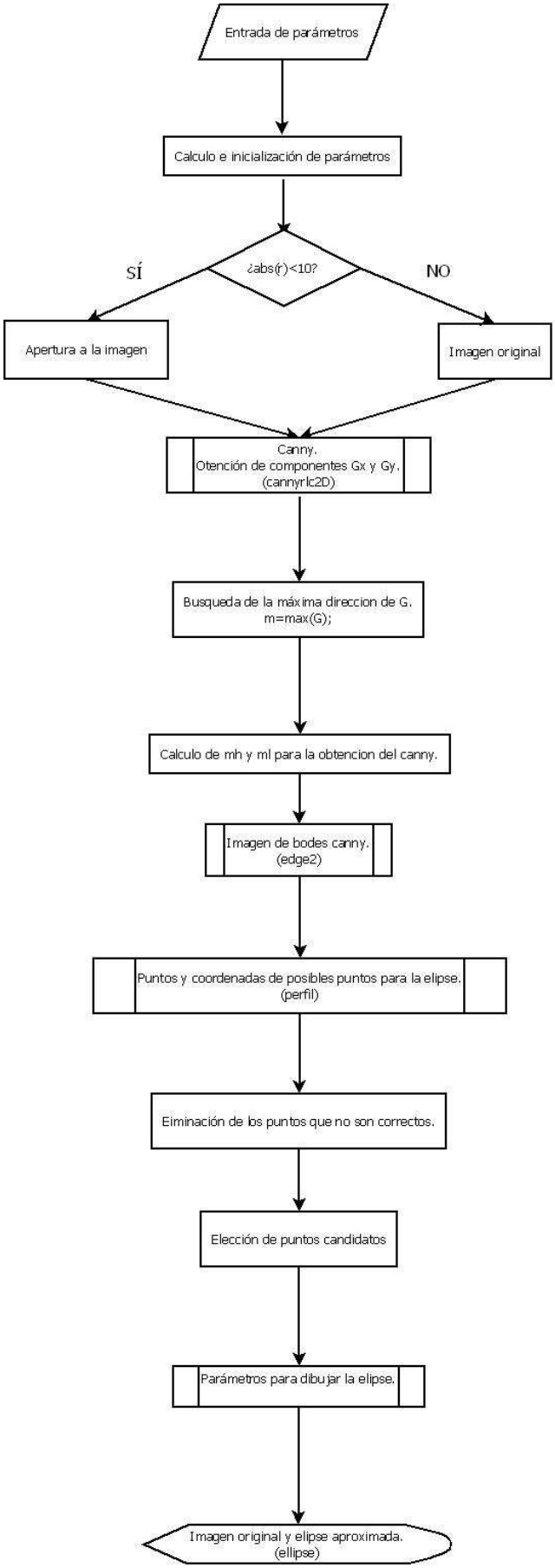


Fig. 10 *Diagrama flujo inicial*

Para la explicación de los siguientes scripts que van a formar el conjunto del programa que consigue la elipse, se adjuntará para cada uno de ellos su diagrama de flujo.

Algunos de los scripts utilizados se han obtenido de otras personas, por lo que se explicará su desarrollo y en las partes que se considere necesario se añadirá diagramas de flujo.

Para seguir un orden, la descripción de los scripts se realizará de acuerdo con su orden de aparición en el programa principal.

3.2.2. Cannyrlc2D

Este algoritmo implementa la primera parte del filtrado Canny (fig. 11). Se trata del cálculo de las componentes vertical y horizontal del gradiente incluyendo el filtrado Gaussiano[1]. Para realizar la llamada a la función utiliza la instrucción

```
[GX,GY]=cannyrcl2D(img,var,var);
```

```
% Parámetros de entrada:
    %I: imagen.
    % sigmax y sigmay: desviación típica para cada una de las direcciones.
% Parámetros de salida:
    % GX y GY: gradiente en cada una de las direcciones.
```

El algoritmo nos va a facilitar los valores de las componentes del filtrado Canny de la imagen. Los parámetros necesarios son la imagen para la cual queremos conocer el gradiente y la desviación típica que queremos que tenga el filtro en cada una de las direcciones.

En este caso se ha elegido la misma desviación típica para ambas direcciones, pero por programación se presenta la posibilidad de poder tener una desviación para cada una de las direcciones.

Se inicializan una serie de parámetros necesarios para el filtrado, se realiza un filtro gaussiano 2D y tras normalizar dicho filtro para que la suma de los coeficientes sea cero se realiza el cálculo de los gradientes en ambas direcciones.

[1] Algoritmo de Rafael Cabeza

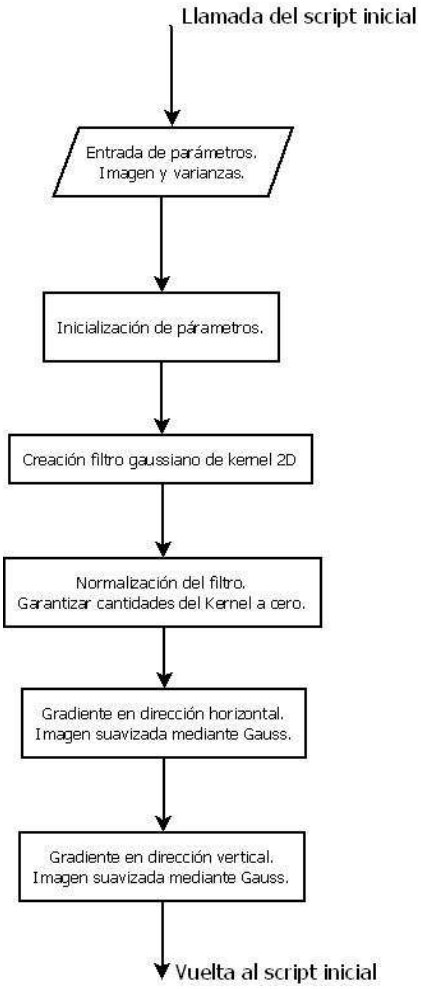


Fig. 11 *Diagrama flujo Cannyrcl2D*

3.2.3. Edge2

La función que se va a utilizar a continuación nos devuelve los bordes de la imagen.

[H,tresh]=edge2(img,'canny',[ml mh],var);

%BW = EDGE(I,'canny',THRESH,SIGMA) specifies the Canny method, using
%SIGMA as the standard deviation of the Gaussian filter. The default
%SIGMA is sqrt; the size of the filter is chosen automatically, based on SIGMA.

Para que el cálculo de la imagen con la representación de los bordes Canny sea más precisa, se ha introducido como datos de entrada una serie de parámetros que nos permiten ajustar mejor la búsqueda, y que el resultado sea más preciso.

La varianza (var) se utilizará para el filtro gaussiano, y el tamaño de la máscara utilizada en el algoritmo será en función de la misma varianza.

A la hora de elegir el umbral, se ha decidido hacerlos relativos al máximo del gradiente obtenido mediante la función cannyrlc2D.

El umbral superior realizará la búsqueda de los bordes fuertes, y el umbral inferior de los bordes débiles. A su vez para que un borde débil sea valido tendrá que cumplir la condición de estar conectado a un borde fuerte.

El algoritmo Canny original está implementado en Matlab, pero el usado ha sido modificado por Rafael Cabeza. La modificación más notable respecto al original en la parte del algoritmo se da en el cálculo del filtro gaussiano, pero la elección del algoritmo modificado frente al de Matlab fue elegido por los resultados que presentaba.

En algunas imágenes no encontrábamos apenas cambio que nos afectara (fig.12), pero en otras (fig. 13), el cambio era bastante notorio en zonas de la imagen que luego afectarían negativamente a los resultados.

Fig. 13. a *Bordes canny edge2*

Fig. 12.b *Bordes canny edge*

Fig. 13.a *Bordes canny edge2*

Fig. 13.b *Bordes canny edge*

3.2.4. Perfil

La siguiente función ayuda a obtener la primera aproximación de los posibles puntos candidatos a pertenecer a la elipse (fig.14).

[posiciones2,corners]=perfil(img,x,y,x1,y1,x2,y2,rx1,ry1,rx2,ry2,flag)

%Parámetros de entrada:
% img: imagen de entrada.
% x e y: Coordenadas que pertenecen al centro de la pupila.
% x1 e y1: Coordenadas que pertenecen al centro de un reflejo
% x2 e y2: Coordenadas que pertenecen al centro de otro de los reflejos.
% rx1 y ry1: Distancias lineales de un reflejo al centro.
% rx2 y ry2: Distancias lineales del otro reflejo al centro.
% flag: Indicador para saber si se ha realizado apertura a la imagen.

% Parámetros de salida:
% posiciones: Array donde se encuentran las distancias donde estan los
% bordes entre iris y pupila, y pupila y esclera respectivamente.
% corners: Array con las coordenadas de la imagen correspondientes a la
% transición del borde perteneciente al limbo.

Como entrada se trabaja con la imagen de la cual queremos obtener esos puntos, las coordenadas correspondientes al centro de la pupila y a los reflejos, distancias de los reflejos al centro de la pupila, y la variable flag.

La variable flag simplemente tiene como utilización para saber si la imagen que se ha introducido se le ha realizado la apertura o no, ya que dependiendo del caso el estudio de puntos será distinto.

El paso de las variables referentes a las distancias lineales entre los reflejos y el centro, rx1, ry1, rx2 y ry2, puede parecer innecesaria, ya que esas distancias son halladas con parámetros que también se pasan a la función, pero es más óptimo pasar los datos calculados, que volver a repetir parte de código para calcular las mismas medidas de nuevo.

En el script es necesario el cálculo de variables, como los ángulos donde se encuentran los reflejos y los signos trigonométricos de dichos ángulos. El algoritmo evita el cálculo de puntos del borde del limbo en aquellos ángulos que contienen a los reflejos. (Matlab usa el sentido horario para la cuenta de los grados (fig.15)).

Una vez que ya se han realizado todos los cálculos, según cuál sea la condición de entrada de la variable, flag, que indica si se ha realizado una apertura en la imagen, se calcularán los rayos para todos los ángulos, o solo para aquellos en los que no se encuentren los reflejos. En ambos casos el cálculo de cada punto del borde correspondiente a cada ángulo se realizará a través de la llamada a la función *rayos*, realizándose el estudio para cada grado. Los resultados de rayos se almacenan en dos vectores como se describe en el siguiente apartado.

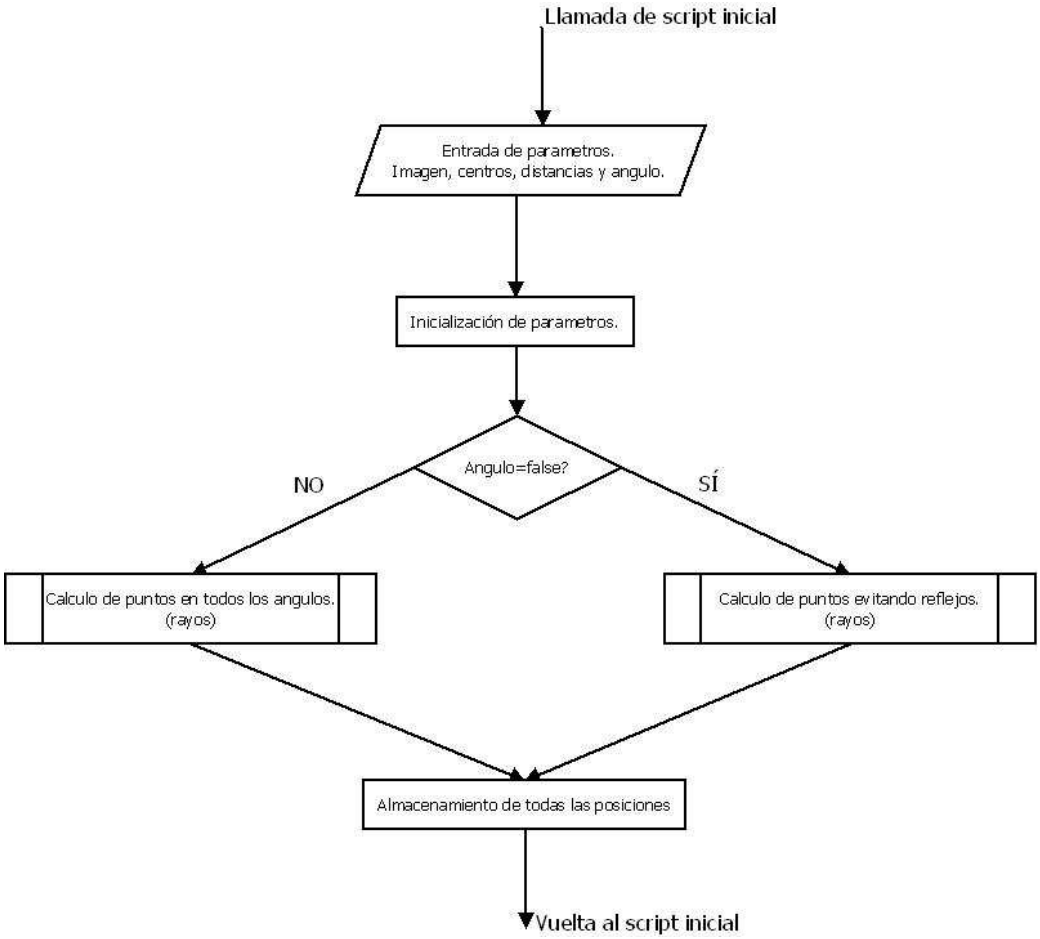


Fig. 14 *Diagrama flujo perfil*

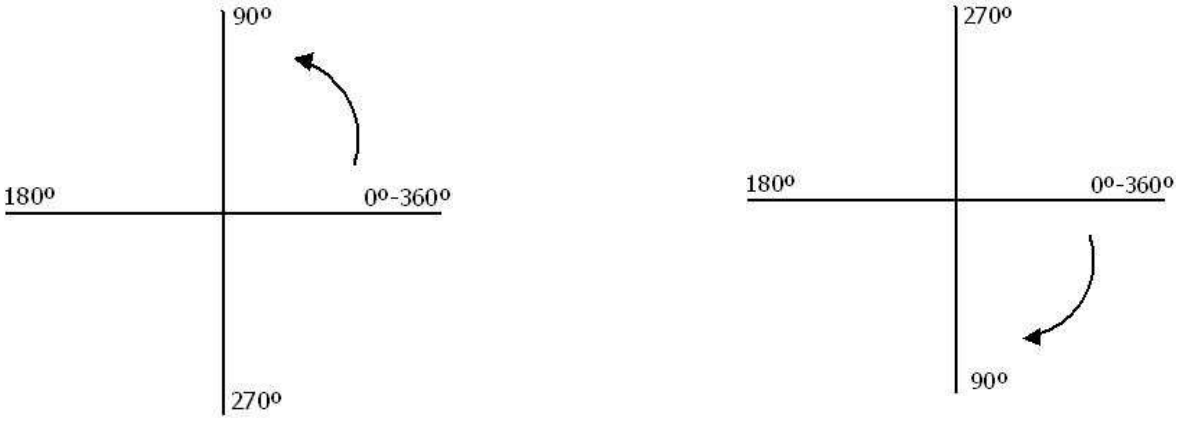


Fig. 15 *Diagrama grados*

3.2.5. Rayos

La función como se ha resaltado anteriormente, calcula la primera aproximación de los posibles puntos para formar la elipse (fig.16).

Como entrada de la función se tiene la imagen de la cual queremos calcular los puntos, que en este caso será la imagen de bordes canny, el ángulo y los puntos de centro y reflejos.

%Parámetros de entrada:
% img: imagen de entrada.
% x e y: Coordenadas que pertenecen al centro de la pupila.
% x1 e y1: Coordenadas que pertenecen al centro de un reflejo
% x2 e y2: Coordenadas que pertenecen al centro de otro de los reflejos.
% fi: ángulo en el que se lanza el rayo.

% Parámetros de salida:
% pos: arrays con las dos posiciones donde se encuentran los
% perfiles.
% corner: Array con las coordenadas de la imagen correspondientes a la
% transición del borde perteneciente al limbo cada ángulo.

Posteriormente se calcularán parámetros iníciales como distancias euclídeas entre reflejos y centro de la pupila, para conocer cuál será el tamaño de los rayos a lanzar.

Inicialmente, dicho parámetro era fijo, pero luego se observó que la necesidad de más o menos puntos por rayo dependía del tamaño del ojo en la imagen por lo que se calcula en función de las distancias, ya que de esta manera independientemente de si la imagen estaba más cercana o alejada siempre se iba a tener la cantidad de puntos necesarios para los posteriores cálculos.

Una vez calculado el tamaño (número de puntos) del rayo se realiza el cálculo de los pixels por donde pasará dicho rayo respecto a la imagen, y sus niveles de gris. El proceso se realizó mediante un proceso bilineal (*improfile*), debido a que la interpolación por este método era la que mejores resultados presentaba.

De esta manera obtenemos el perfil de intensidades del rayo sobre el que buscaremos picos mediante funciones de Matlab (*findpeaks)*. Para luego quedarnos con el segundo pico encontrado por la función.

Asumiendo que el primer pico representa el contorno de la pupila y el segundo el del limbus, se debe tener en cuenta que dicha función (*findpeaks)* no da siempre dos puntos, que sería lo ideal según se ha planteado el problema, por lo que para ello se estudiarán todos los casos posibles.

Lo primero que se debe hacer es comprobar si el número de picos encontrado ha sido dos o superior. En este caso se volverá a realizar la búsqueda forzando a la función (*findpeaks)* a encontrar dos picos solamente y almacenando dichas posiciones.

Si el resultado solo ha sido de un pico se almacenará dicha posición más NaN, para así saber que solo ha encontrado un pico. En el caso de que no se encuentre ningún pico el resultado almacenado será de NaN en las dos posiciones.

En el caso de que se hayan encontrado dos o más picos se calculan las coordenadas correspondientes al punto en la imagen original. Si no se han encontrado dos picos, el resultado de las coordenadas es cero.

De esta manera al finalizar la función *rayos* se vuelve a la función *perfil* con las coordenadas del segundo pico encontrado correspondiente a la imagen original, o con ningún pico si no se han encontrado dos picos.

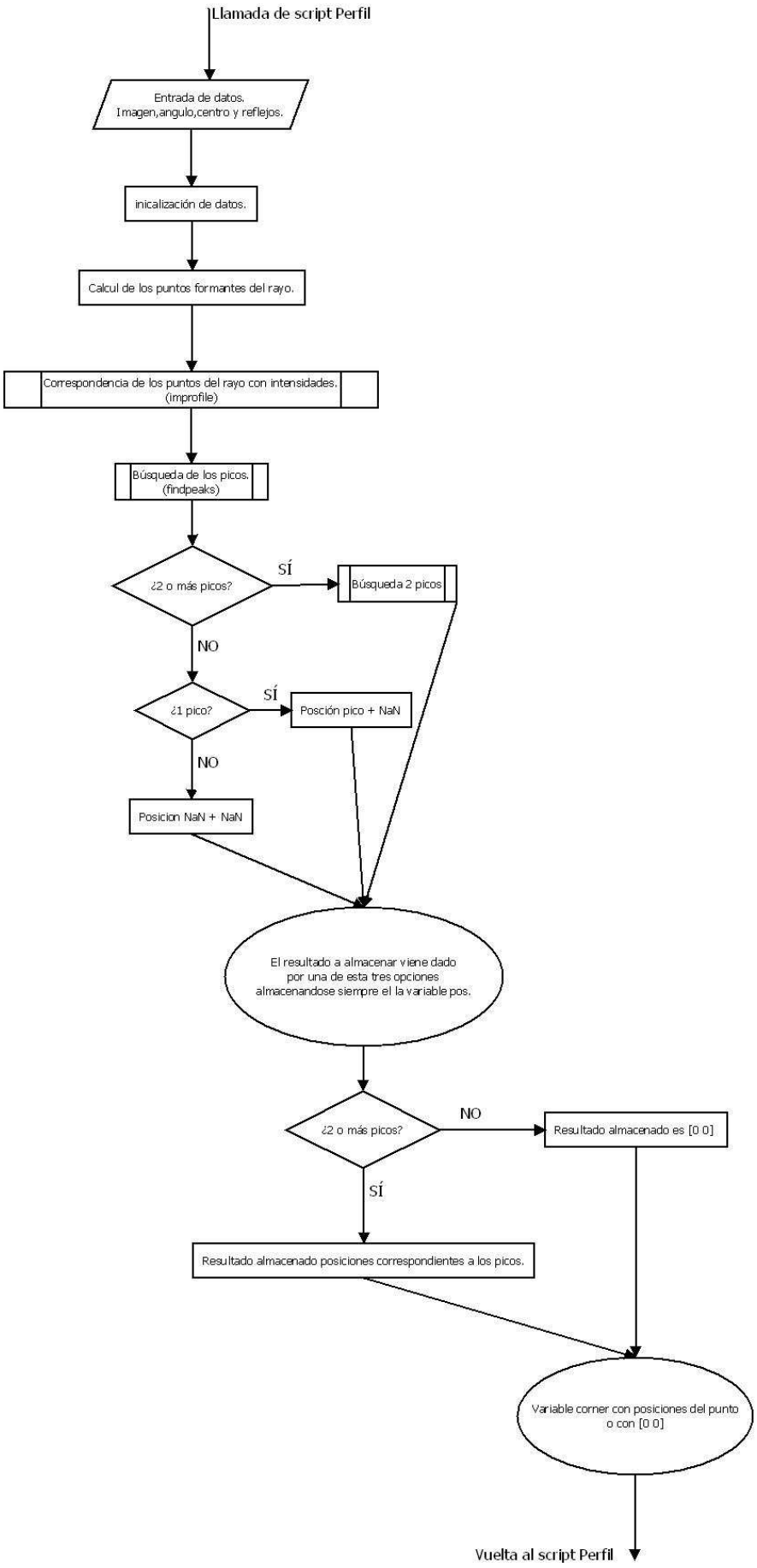


Fig. 16 *Diagrama flujo rayos*

Una vez obtenidos todos los puntos que se requieren para la aproximación a la elipse, se necesitan calcular unos parámetros necesarios.

Estos parámetros a calcular son el centro de la elipse, los semiejes tanto mayor como menor y el ángulo de orientación.

Para ello se va a explicar los dos métodos por los cuales se han obtenido dichos puntos.

3.2.2 Elección de puntos

3.2.2.1 Ellipse_fit

La primera aproximación de la valores se realizará mediante el uso de la función ellipse_fit.[2]

Los parámetros de entrada serán dos vectores, que en cada uno de ellos se encontraran las coordenadas *x* e *y* pertenecientes a los puntos.

Como salida del script se obtendrán los parámetros que son necesarias para dibujar la elipse, que son el centro, los semiejes mayor y menor y la orientación.

```
% Input:
        %x - a vector of x measurements
        %y - a vector of y measurements

% Output:
        %semimajor_axis - Magnitude of ellipse longer axis
        %semiminor_axis - Magnitude of ellipse shorter axis
        %x0 - x coordinate of ellipse center
        %y0-  y coordinate of ellipse center
        %phi - Angle of rotation in radians with respect to the x-axis
```

Para el cálculo correcto de la elipse, el algoritmo intenta encontrar la mejor aproximación haciendo uso de la ecuación cuadrática de la elipse. Para ello al tener varios parámetros que desconocemos en dicha ecuación, divide la ecuación en dos partes, y tras realizar transformaciones matemáticas, realiza el cálculo de todos los parámetros desconocidos.

Cuando todos los parámetros de la ecuación cuadrática de la elipse son hallados correctamente, se realizan otra serie de ecuaciones que ayudan al cálculo de los parámetros de salida al que se aproxima la elipse.

[2] Creada por Hendel. Faculty of Biomedical Engineering, Technion- Israel Institute of Technology

3.2.2.2 Algoritmo Ransac

El segundo cálculo de los valores relevantes de la elipse, se ha realizado mediante un algoritmo tipo *ransac*.

Ransac es una abreviatura de "Muestra aleatoria Consenso". Se trata de un método iterativo para estimar los parámetros de un modelo matemático de un conjunto de datos observados, que contiene los valores extremos

Es un algoritmo no determinista en el sentido de que produce un resultado razonable sólo con una cierta probabilidad, con esta probabilidad cada vez mayor a medida que más iteraciones se admiten.

Un supuesto básico es que los datos se componen de “inliers”, es decir, datos cuya distribución se explica por un conjunto de parámetros del modelo, y "atípicos", que son datos que no encajan en el modelo. Además de esto, los datos pueden estar sujetos al ruido.

Los valores extremos se pueden formar, por ejemplo, de los valores extremos del ruido o de mediciones erróneas o hipótesis incorrectas sobre la interpretación de los datos.

Ransac también asume que, dado un conjunto (por lo general pequeño) de “inliers”, existe un procedimiento que puede estimar los parámetros de un modelo que explica de manera óptima o se ajusta a esta información.

Parte de las funciones necesarias se han extraído del paquete *Starburst.* Siendo *Starburst* un algoritmo de libre distribución para Eye-tracking basado en pupila y reflejo, los cuales son *fit_ellipse_ransac*, *normalize_point_coordinates*, *convert_conic_parameters_to_ellipse_parameters* y *denormalize_ellipse_parameters*[3].

La función principal a la que se hace referencia en el script inicial es *fit_ellipse_ransac*, y a partir de la cual va haciendo uso de las otras funciones mencionadas. En el uso de ella sobre nuestro algoritmo se detectaron algunos errores, los cuales llevaban a la problemática de que cada vez que se ejecutase el algoritmo los resultados de la mejor elipse variaban.

El error más grave encontrado fue al analizar el procedimiento de los valores de la elipse. Se realizaban subconjuntos de cinco puntos, a partir de los cuales encontraba una elipse, y realizando otros subconjuntos calculaba los puntos que podían pertenecer a dicha elipse, sin embargo el cálculo de los parámetros de la elipse solo los calculaba con los cinco primeros puntos.

No se realizaba una comparación entre las mejores elipses que iba obteniendo ni un promedio de ellas, por lo que se realizaron algunos cambios sobre la base del algoritmo que se disponía.

[3] Creado por Dongheng Li y Derrick Parkhurst.

Se forzó al algoritmo a realizar un número fijo de iteraciones para encontrar la mejor elipse, a tener una cantidad mínima de inliers (puntos que nos dan los parámetros que forman la mejor elipse) y que el resultado final de la elipse optima fuera aquella que más inliers contuviera[4].

Por lo que tras todo el análisis tenemos como punto de partida la llamada a función con la instrucción

```
[max_ellipse, max_inlier_indices, ransac_iter] = fit_ellipse_ransac(x, y, maximum_ransac_iterations);

% Input
    % [x,y] = feature points (row vectors)
    % maximum_ransac_iterations

% Output
    % max_ellipse = best fitting ellipse parameters
    % max_inlier_indices = inlier indices for max_ellipse
```

Donde x e y son los vectores con los puntos finales candidatos para formar la elipse, y maximun_ransac_iterations nos permite introducir el número máximo de iteraciones que queremos que el algoritmo realice para encontrar la mejor elipse.

Como salida de la función obtenemos los parámetros necesarios para formar la elipse en max_ellipse, y a la vez también podemos conocer las iteraciones que ha realizado antes de encontrar la elipse adecuada y los puntos que utilizada para su cálculo.

3.2.3 Ellipse

Al contar con los parámetros de la elipse, se podría dar por finalizado dicho algoritmo.

Mediante la entrada de los parámetros calculados por uno de los dos métodos anteriores, se realiza el dibujo de la elipse que estamos buscando sobre la imagen original con la ayuda de la función *ellipse*.[5]

```
% ELLIPSE(ra,rb,ang,x0,y0) adds an ellipse with semimajor axis of ra, a semimajor
axis of radius rb, a semimajor axis of ang, centered at the point x0,y0.
```

El algoritmo utiliza las dimensiones de los parámetros de entrada para el posterior cálculo de los datos necesarios para el dibujo. De esta manera mediante cálculos trigonométricos, funciones ya definidas por Matlab y los datos de entrada se construye el dibujo.

[4] Cambios realizados por Arantxa Villanueva Larre.

[5] Creada por D.G. Long, Brigham Young University y Peter Blattner, Institute of Microtechnology, University of Neuchatel

4 RESULTADOS Y COMPARATIVA

4.1 PRÓLOGO

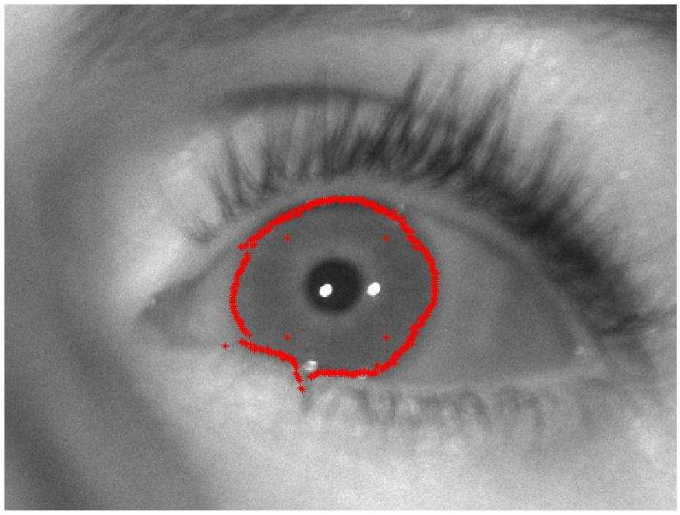
Para el cálculo de la elipse se presentan tres posibilidades que se detallan a continuación.

La primera de ellas se basa en realizar el cálculo de la elipse con todos los puntos candidatos obtenidos. Por otro lado tenemos la elección de los puntos que cumplen la característica de encontrarse dentro de un umbral. Y por último la elección de puntos según su gradiente en la dirección vertical.

4.1.1 Todos los puntos candidatos

Al hablar de todos los puntos candidatos, nos referimos a aquellos puntos que tras ser calculados en la imagen Canny, han pasado el filtro de no ser NaN.

Para ello observaremos los resultados obtenidos mediante este método, viendo para distintos casos de imágenes que puntos da como resultado (fig.17).



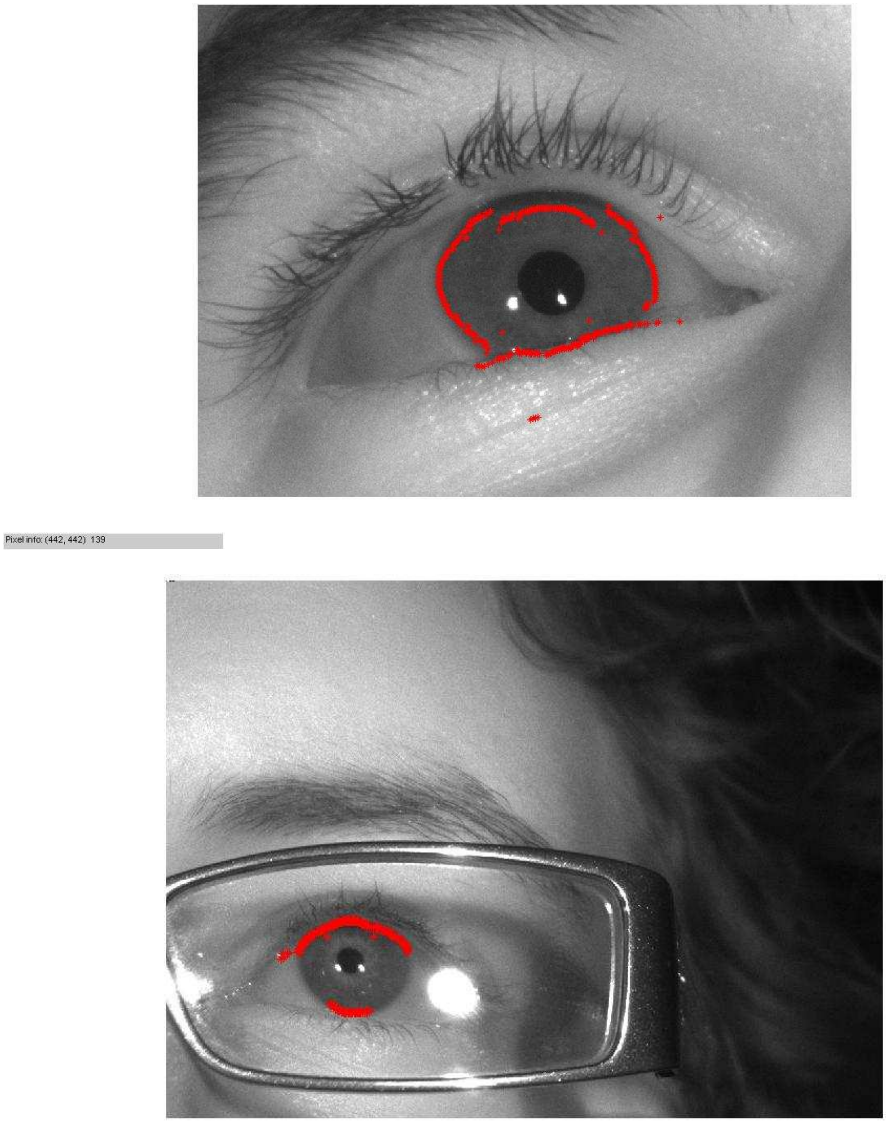


Fig. 17 *Ejemplos de todos los puntos candidatos*

4.1.2 Elección de puntos candidatos por umbral

Otro de los procedimientos propuestos para la elección de puntos es mediante un umbral (fig18).

Dicha elección se basa en la distancia de los puntos al centro de la pupila. En este caso se determina un umbral, que será definido a partir de la media de todas las distancias y su desviación típica.

Si la distancia de cada punto con respecto al centro se encuentra dentro del umbral el punto será válido, si no se eliminará.

El umbral al que nos referimos será determinado en su parte inferior por la *media - desviación típica* y en la parte superior por la *media + desviación típica*.

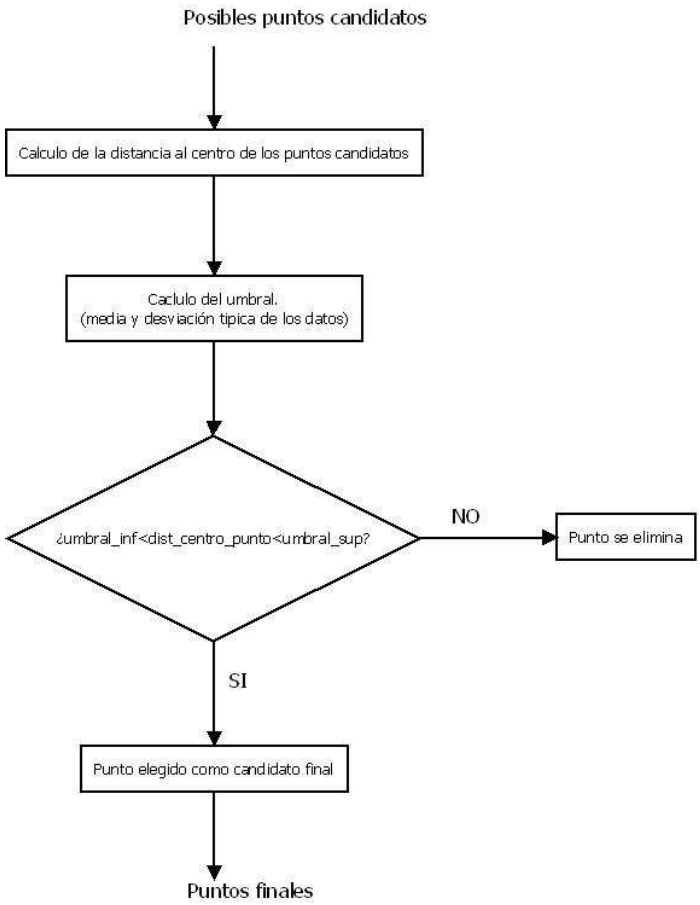


Fig. 18 *Diagrama flujo elección umbral*

Para observar cómo actúa dicho umbral tras la elección de los puntos primeros se muestra para la misma imagen todos los puntos, y después aquellos que han pasado el umbral marcado (fig.19).

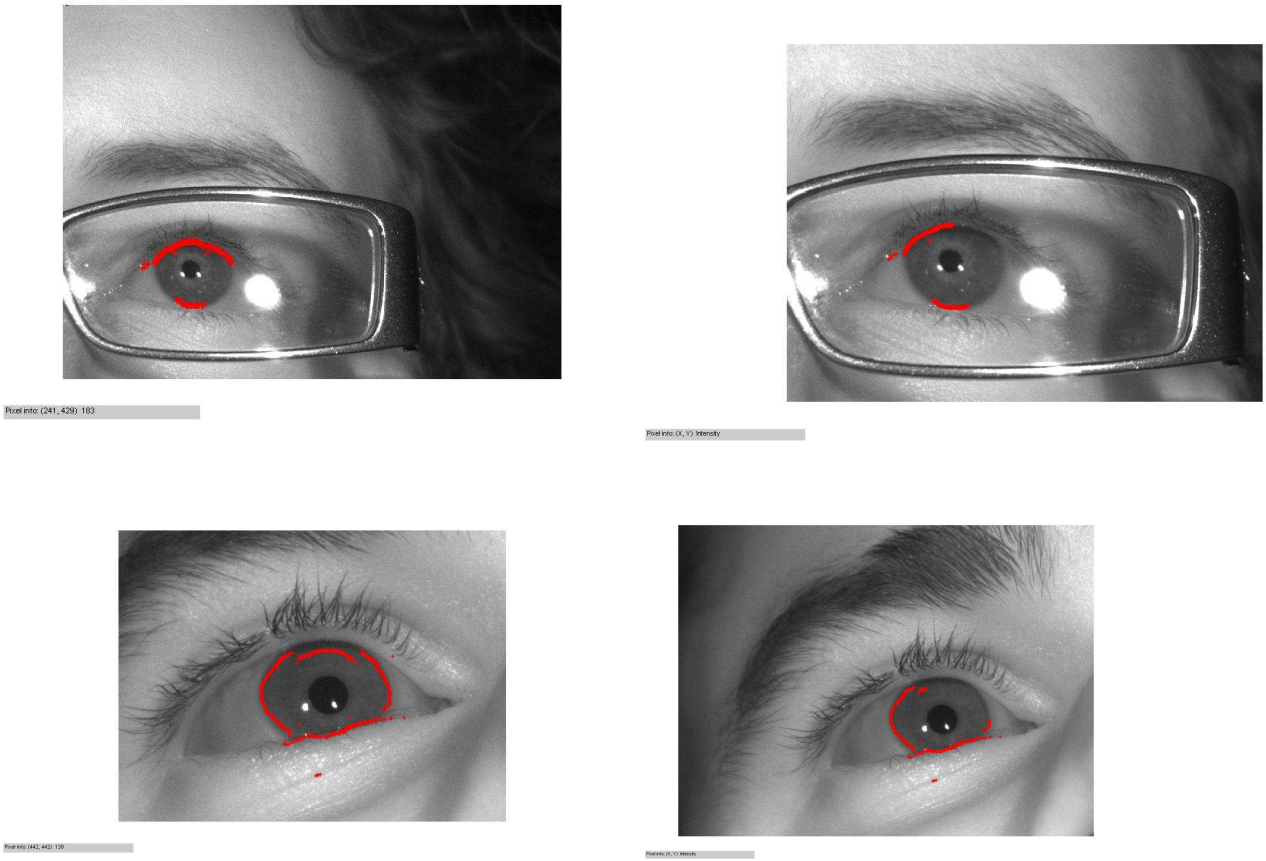


Fig. 19 *Ejemplos de la selección de puntos tras la elección umbral*

4.1.3 Elección de puntos candidatos por gradiente.

El último procedimiento, elección de puntos por gradiente (fig.20), se realizó mediante el uso de la imagen gradiente obtenida mediante la función *cannyrlc2D*.

De esta función se ha de recordar que se obtenían los resultados en las dos componentes de la imagen, tanto en x como en y.

Para la correcta utilización se trabajó en este caso con los resultados, pertenecientes a la componente vertical de la imagen. De esta manera tenemos la certeza de que los máximos se encontrarán en las transiciones de los bordes verticales.

Para ello buscamos la coincidencia de los segundos picos obtenidos de la imagen canny, con los segundos picos de la imagen gradiente vertical.
En el caso idóneo de la imagen gradiente, el primer pico resultante tiene que pertenecer a la transición entre la pupila y el iris, y el segundo pico a la transición de la pupila y la esclera.

Ello es debido a que los máximos del gradiente en la componente vertical se encuentran en las transiciones descritas.

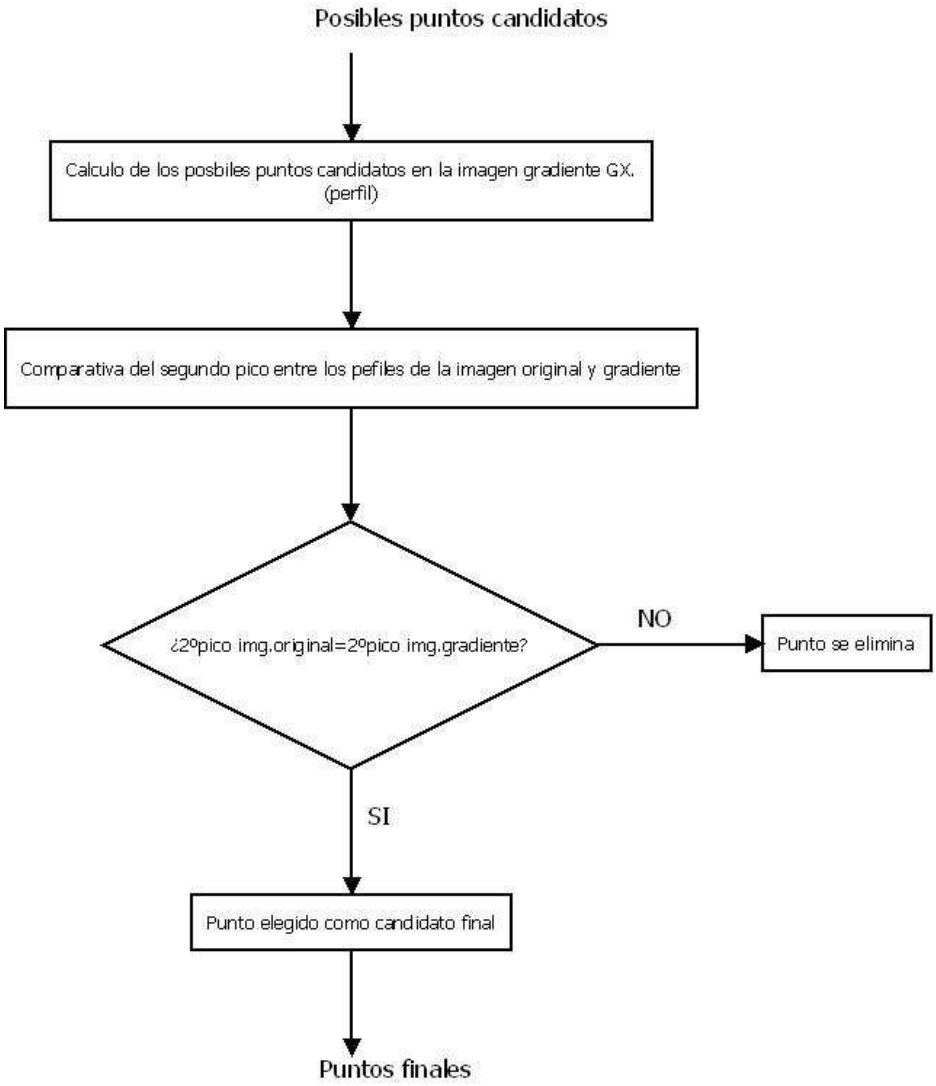
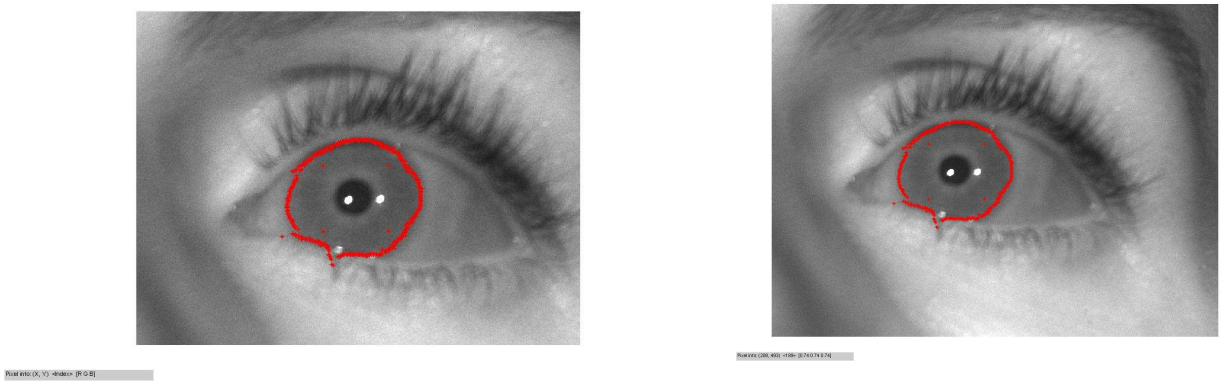


Fig. 20 *Diagrama flujo elección gradiente*

Para ello la comparación respecto a todos los puntos que se obtienen es adecuada para observar las diferencias y las restricciones que dicho método presenta (fig.21).



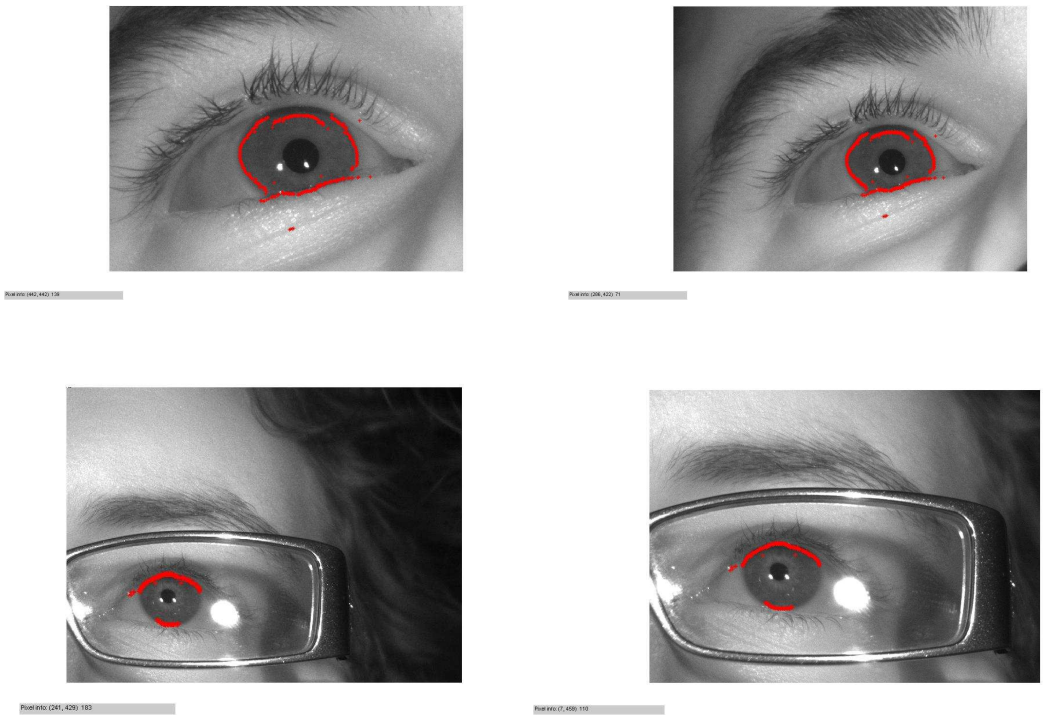


Fig. 21 *Ejemplos de selección puntos tras elección gradiente.*

4.2 MÉTODOS

Una vez descritos los tres tipos de algoritmos utilizados para la detección de puntos, se explicará las combinaciones utilizadas entre ellos y los algoritmos que calculan los valores de la elipse.

Se pasarán a exponer los diferentes tipos de métodos evaluados, realizando una comparación mediante error. Dicho error será calculado para todos los métodos mediante la fórmula de la distancia algebraica.

Para poder calcular el error comentado, se realizaron previamente una serie de pasos.

En primer lugar ha de señalarse que los resultados se analizan sobre un total de 181 imágenes diferentes, capturas para distintos puntos de la mirada, de diferentes sujetos y de ambos ojos por separado.

Los sujetos varían alrededor de cinco, de los cuales se ha tenido en cuenta que entre ellos se tuviera la diversidad de que utilizaran lentillas, gafas, que fueran de ojos claros y oscuros, y tanto hombres como mujeres.

Posteriormente dos sujetos distintos realizaron una marcación de puntos sobre el total de las imágenes. Esta marcación se realizó en el borde entre el iris y la esclera, con la que finalmente se obtuvieron unos puntos, de los cuales se obtienen los datos de la *elipse_true*.

Dicha *elipse_true* se define de tal manera, que es con la que vamos a comparar el error, ya que nos da los valores de la elipse más óptimos que se pueden conseguir. Estos valores están calculados con el conjunto de la marcación de los puntos de ambos sujetos.

Por lo que una vez que se tienen los valores de la *elipse_true*, se utiliza la formula de la distancia algebraica para conocer el error existente entre la mejor elipse y la aproximación que se realiza por cada método.

Como los algoritmos que componen todos los métodos están explicados ampliamente con anterioridad, se comentará la combinación realizada y el nombre asignado a cada uno de ellos, para que en las comparaciones de resultados, todos los nombres se relacionen sin problema alguno (tabla 1).

	Todos	**Umbral**	**Gradiente**
Ransac	todos_ransac	umbral_ransac	gradiente_ransac
Ellipse_fit	todos_fit	umbral_fit	gradiente_fit

Tabla 1. *Combinación de métodos*

El **gradiente_fit** estará formado por la elección de puntos mediante gradiente, y la búsqueda de los parámetros de la elipse por *ellipse_fit* (fig.22). Por otro lado con el mismo algoritmo para la elección de puntos, pero utilizando para la búsqueda de parámetros el algoritmo *ransac* tenemos el **gradiente_ransac** (fig.23). Como se puede observar hay casos en los que el uso del *ransac* mejora la elipse óptima (fig.24), frente al uso de *ellipse_fit* (fig.25).

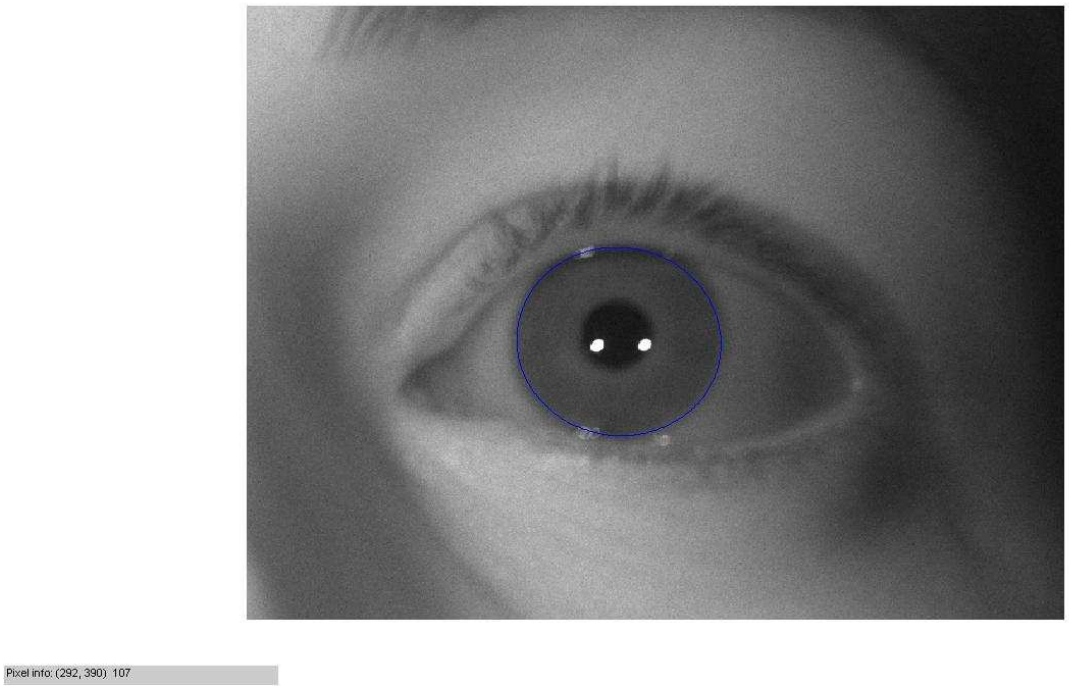


Fig. 22 *Ejemplo elipse por* ***gradiente_fit***

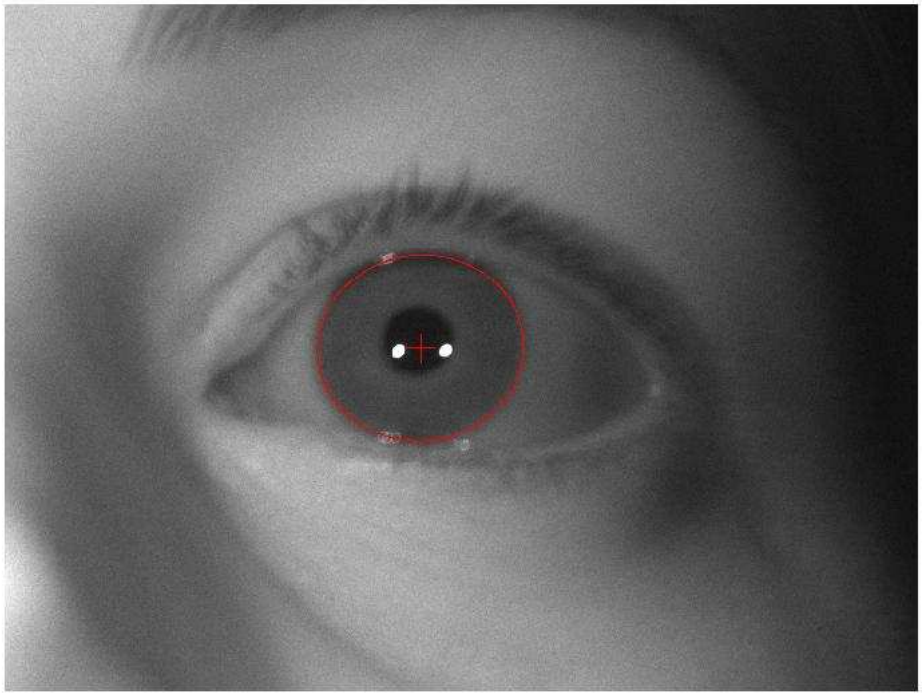
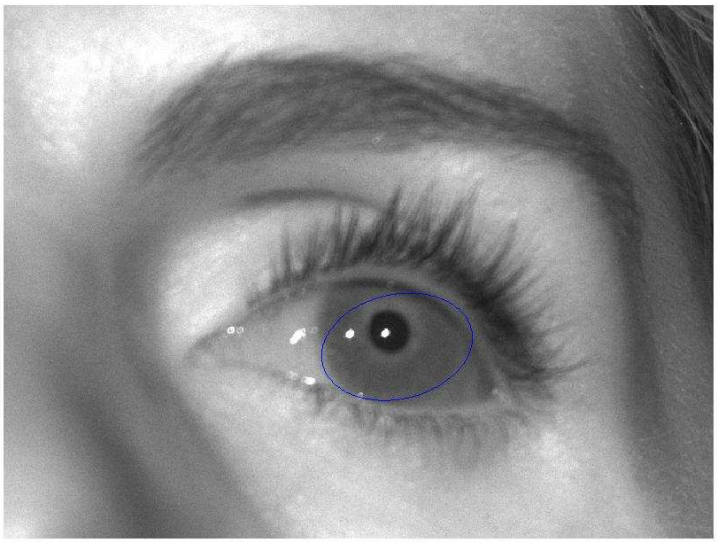


Fig. 23 *Ejemplo elipse por* ***gradiente_ransac***

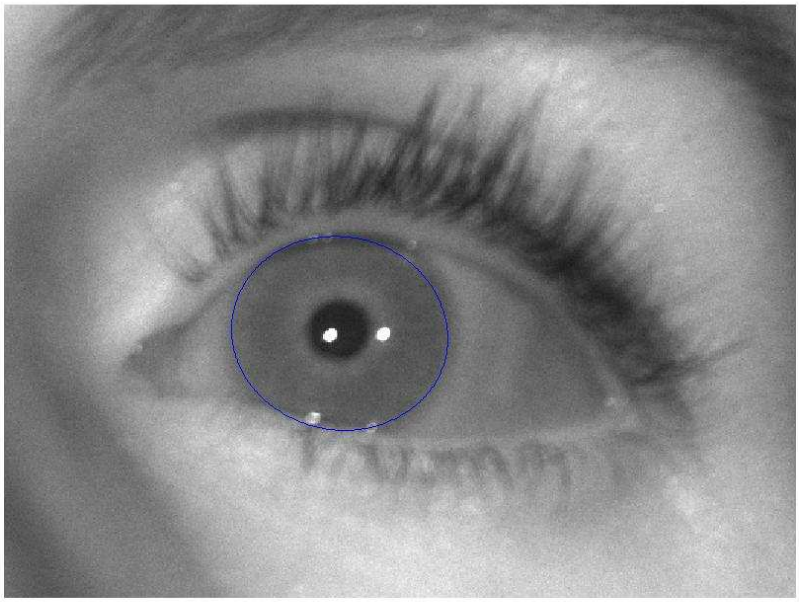
Fig. 24 *Ejemplo elipse por* ***gradiente_ransac***



Pixel info: (X, Y) <Index> [R G B]

Fig. 25 *Ejemplo elipse por* ***gradiente_fit***

El método **todos_fit** lo formarán todos los puntos candidatos y el algoritmo *ellipse_fit* (fig.26). Por otro lado para todos los puntos y el algoritmo *ransac* nos referiremos con el nombre de **todos_ransac** (fig.27).



Pixel info: (130, 518) <109> [0.43 0.43 0.43]

Fig. 26 *Ejemplo elipse por* ***todos_fit***

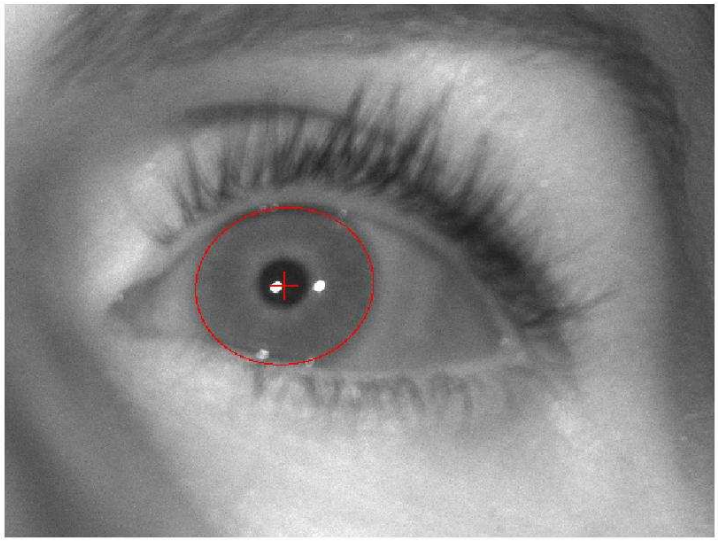


Fig. 27 *Ejemplo elipse por* ***todos_ransac***

Para la finalización de estos algoritmos se tendrán **umbral_fit** (fig.28), que tiene como elección de puntos por umbral y utiliza el algoritmo *ellipse_fit*, y **umbral_ransac** (fig.29), que con la misma elección de puntos pero con el algoritmo *ransac*.

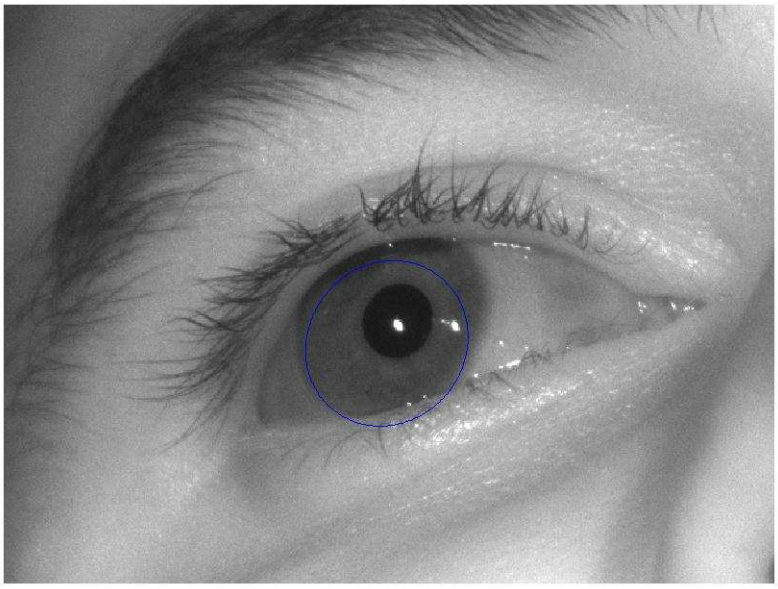


Fig. 28 *Ejemplo elipse por* ***umbral_fit***

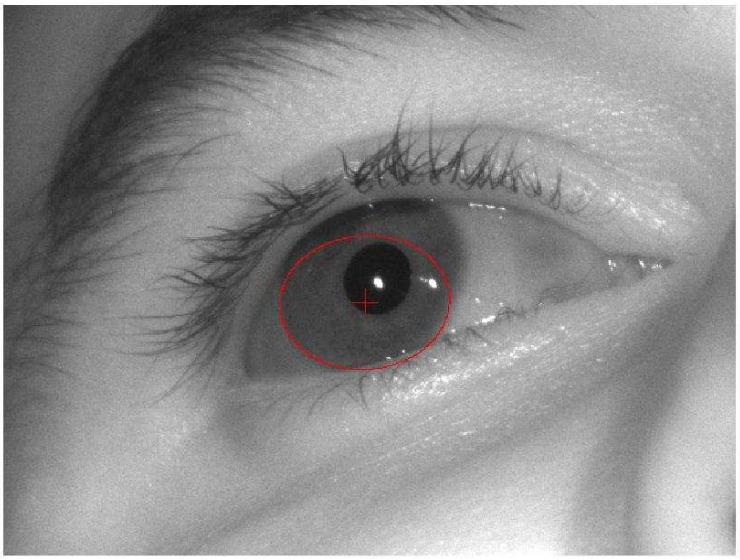


Fig. 29 *Ejemplo elipse por* ***umbral_ransac***

4.3 ANÁLISIS

Conforme se fueron realizando y analizando estos procedimientos, se observó que alguno de ellos mostraba algunas carencias. Dichas carencias se producían por diversos factores, por lo que se optó por realizar un último método llamado *hibrido*, que se explica más adelante.

Para la elección o decisión del algoritmo más óptimo se han calculado los errores de cada uno de los métodos. Dichos errores son calculados mediante la ecuación de la distancia algebraica, comentado anteriormente.

Para decidir correctamente se ha mostrado la comparación de los errores medios de todas la imágenes con sus desviaciones típicas (fig.30), teniendo que mostrar a su vez por separado los datos pertenecientes a los métodos que utilizan *ransac*, ya que si no la apreciación de ellos no sería tan clara (fig.31). Al igual el error máximo y mínimo obtenido en cada método (fig 32 y fig.33).

Por si visualmente no se aprecia correctamente, se adjuntan los resultados obtenidos numéricamente (tabla 2).

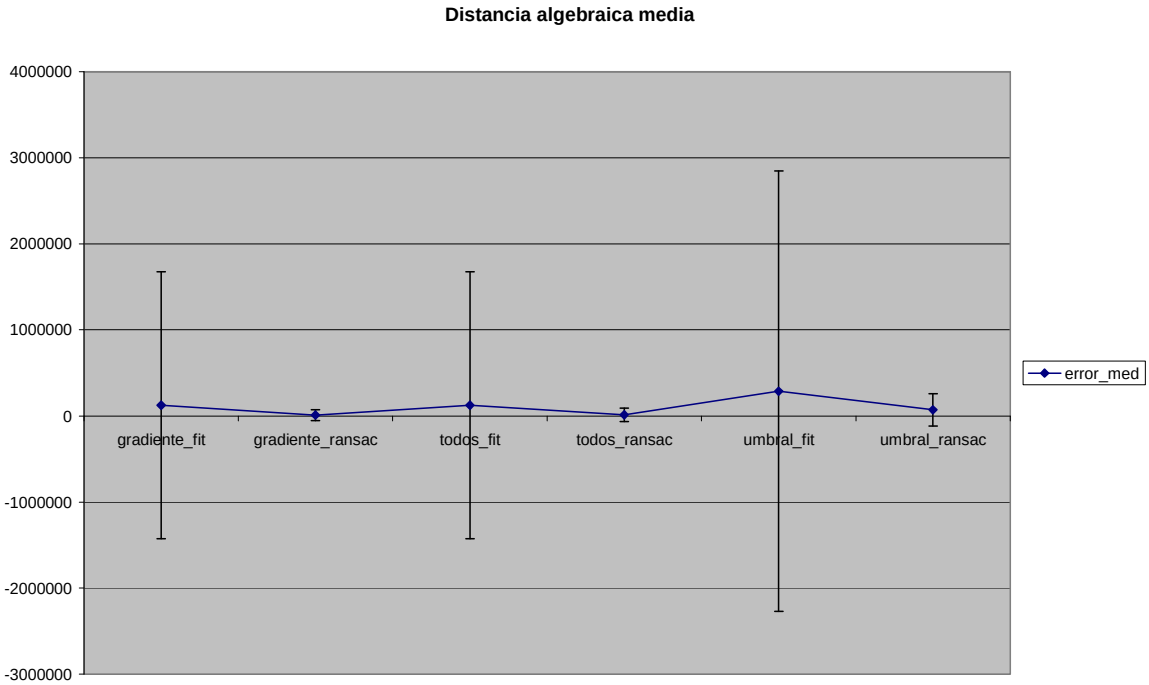


Fig.30. *Distancia algebraica media y desviaciones típicas*

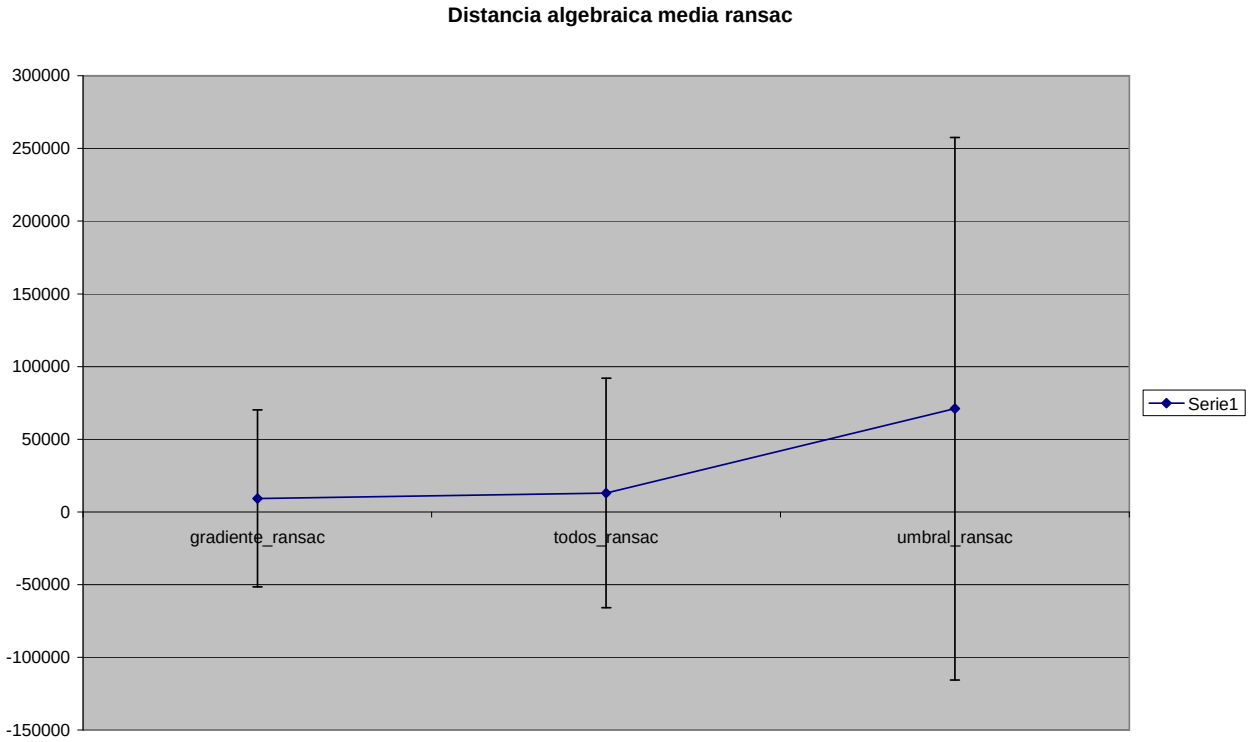


Fig.31. *Distancia algebraica media y desviaciones típicas ransac*

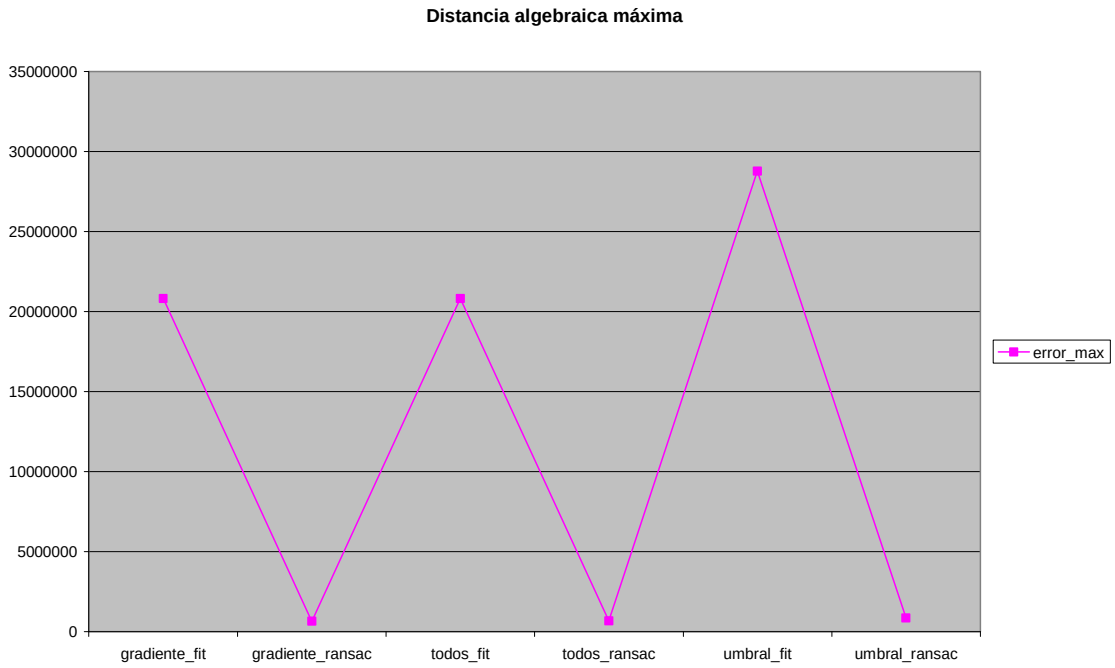


Fig.32. *Distancia algebraica máxima.*

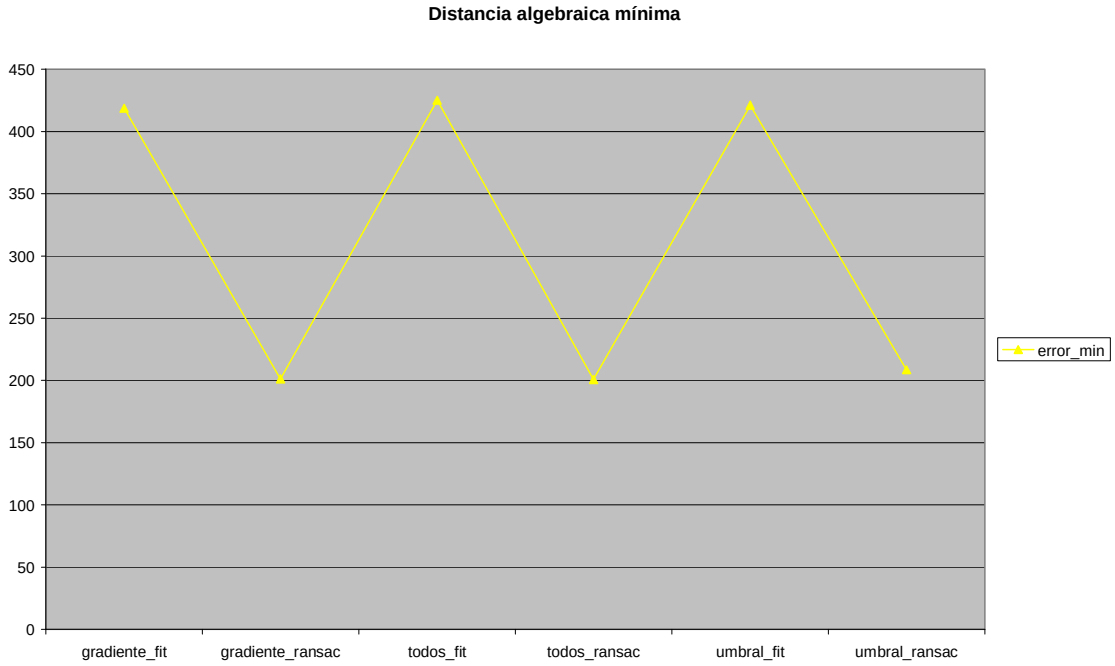


Fig.33. *Distancia algebraica mínimas.*

	error_med	error_max	error_min
gradiente_fit	123745,5653	20831981,39	418,8308479
gradiente_ransac	9358,755097	656139,5573	201,3228313
todos_fit	123753,1595	20833381,11	425,2827754
todos_ransac	13113,15837	684285,4127	200,8299002
umbral_fit	286696,6814	28788628,43	421,168555
umbral_ransac	71078,55248	850948,3883	208,7006863

Tabla2. *Distancias algebraicas*

Con estos datos se puede apreciar como se ha comentado anteriormente, que en aquellos métodos en los cuales se utilizada el algoritmo *ransac* para el cálculo de los parámetros de la elipse, los resultados mejoran (fig.30 y tabla 2).

Para concluir un mejor método se podría discutir el error entre el *gradiente_ransac* y *todos_ransac* (fig.34). Aunque por los resultados mostrados por las distancias algebraicas, el algoritmo *gradiente_ransac* favorece notablemente a los resultados respecto al *todos_ransac.*

Refiriéndonos a los datos mostrados en la tabla anterior según que parámetro de error se observase un método es mejor que otro. Por lo que para ello, hay que observa la grafica, donde se muestran los errores de ambos métodos para cada una de las imágenes simuladas (fig 35).

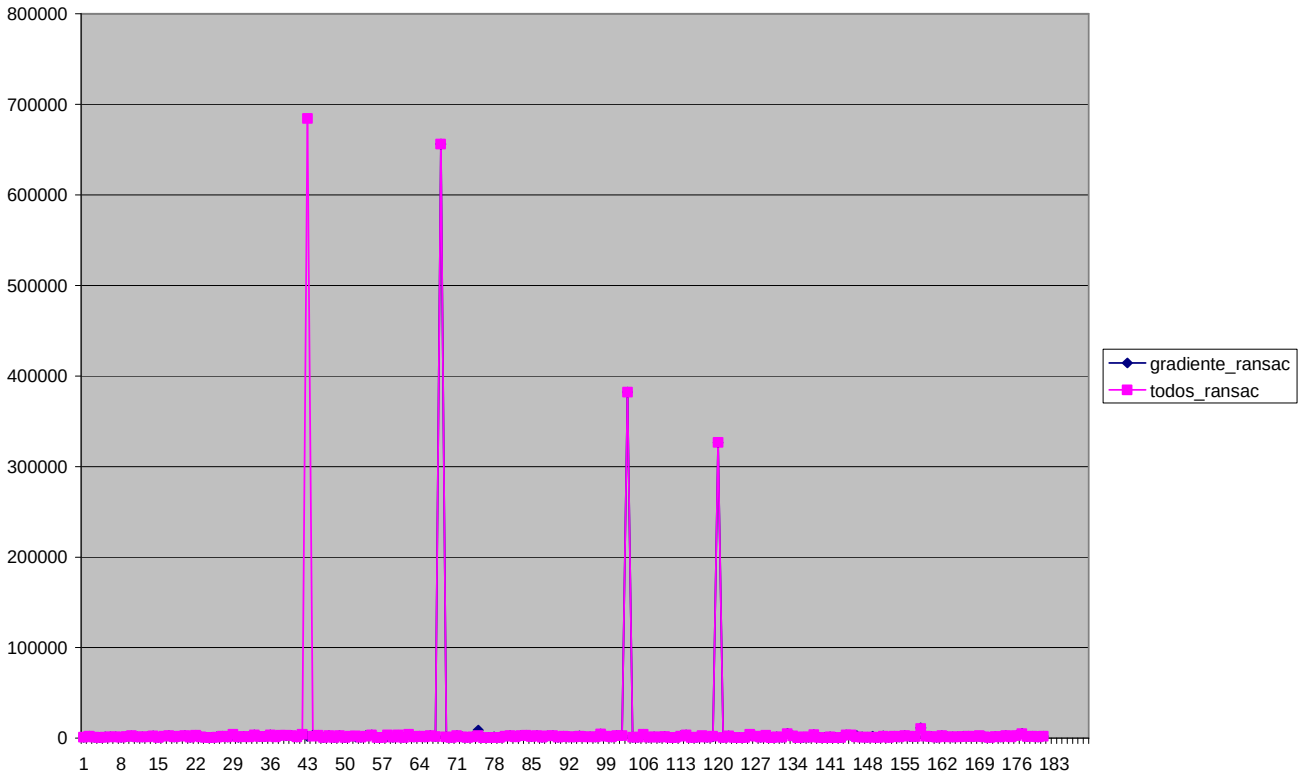


Fig. 34. *Distancias algebraicas de cada imagen*

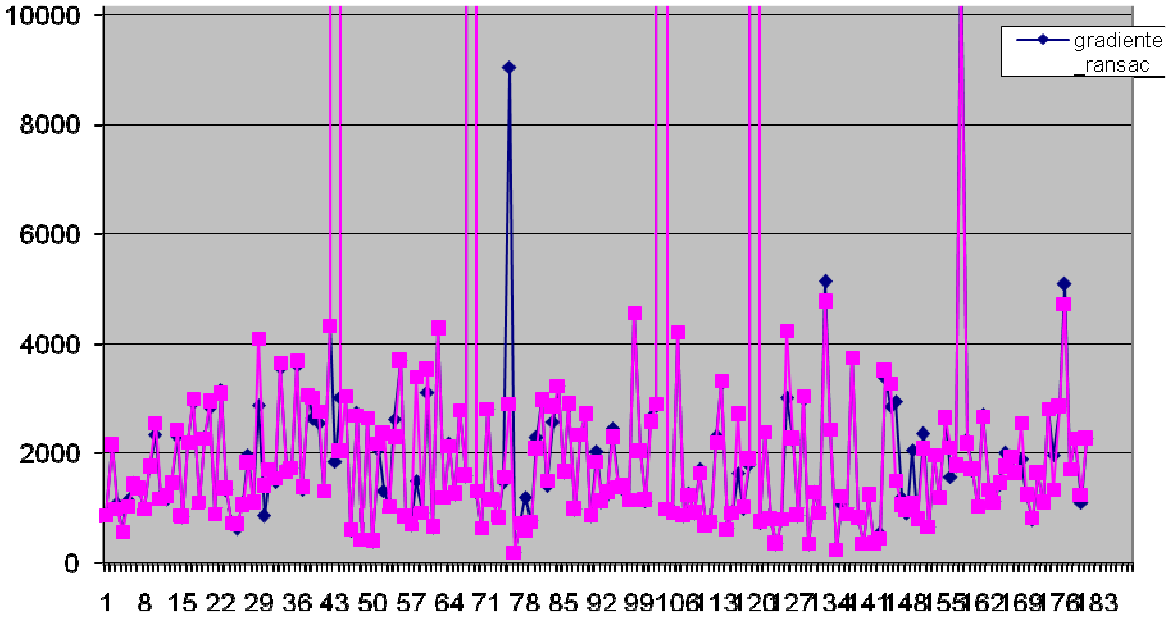


Fig. 35. *Distancias algebraicas de cada imagen*

Viendo los resultados no se puede concluir que un método mejore respecto al otro. Quizá un test de comparación estadístico podría arrojar algo de luz en la discusión.

Una vez analizados los sistemas entre sí, y vista la comparación de los errores que presentan, se realizará la comparación con el sistema utilizado.
Ello es debido a que el cálculo y la comparación de las distancias algebraicas nos permiten la comparación entre los métodos, pero no es representativa de la bondad de los resultados.

Para ello se ha realizado una simulación con un modelo geométrico virtual 3D. En el cual podemos fijar las características del ojo, y calcular el centro del iris, la elipse de la transición entre iris y pupila, la elipse entre el iris y la esclera y la situación de ambos reflejos.

A partir de estos datos, y mediante un sistema de estimación de la mirada geométrico se conoce el punto de la pantalla al que está mirando el ojo. Mediante el modelo geométrico, se ha medido la sensibilidad del método de estimación de la mirada a la precisión de los diferentes parámetros de la elipse calculada.

Para evaluar el sistema respecto a los resultados obtenidos se ha tenido en cuenta que el error del punto de la pantalla al que está mirando el ojo sea inferior a un grado. El estudio teórico realizado no es completo. Se ha realizado para el punto central de la pantalla, por lo que las conclusiones respecto a la sensibilidad son aproximadas.

Conociendo *a priori* los parámetros ideales de la elipse cuando el ojo está mirando a ese punto, y teniendo en cuenta el grado de error del punto de la pantalla, la sensibilidad respecto a las coordenadas del centro tiene que ser menor a 20 pixels en la imagen de trabajo, la del ángulo de orientación menor al 10% y la de la excentricidad al 9%.

Asumiendo que esta sensibilidad está calculada únicamente para un solo punto de la pantalla, los datos estadísticos son orientativos.

A pesar de que anteriormente se han evaluado los métodos que comprendían los errores más pequeños en la simulación, no se va a dejar de medir las sensibilidades descritas respecto al modelo virtual para aquellos métodos en los que el error era mayor (tabla 3).

Teniendo en cuenta los porcentajes que tienen que cumplir las diferencias entre las elipses, observemos para cuantos casos en cada método se cumplen dichas necesidades (tabla 4).

	centro_x	centro_y	phi	e	todos
gradiente_fit	169	159	12	30	5
gradiente_ransac	165	159	16	151	13
todos_fit	169	159	12	30	5
todos_ransac	166	159	13	148	11
umbral_fit	137	125	5	24	2
umbral_ransac	137	139	7	150	6

Tabla3. *Nº imágenes que cumplen la sensibilidad*

	centro_x	centro_y	phi	e	todos
gradiente_fit	93,3701657	87,8453039	6,62983425	16,5745856	2,76243094
gradiente_ransac	91,160221	87,8453039	8,83977901	83,4254144	7,18232044
todos_fit	93,3701657	87,8453039	6,62983425	16,5745856	2,76243094
todos_ransac	91,7127072	87,8453039	7,18232044	81,7679558	6,07734807
umbral_fit	75,6906077	69,0607735	2,76243094	13,2596685	1,10497238
umbral_ransac	75,6906077	76,7955801	3,86740331	82,8729282	3,31491713

Tabla4. *% imágenes que cumplen la sensibilidad*

Tiene que quedar claro, que el número de aciertos de cada caso no tiene que corresponder a una misma imagen. Pongamos el ejemplo para el *gradiente_fit*, las doce imágenes que cumplen el criterio del error del ángulo de orientación *(phi)* no tienen por qué cumplir también los otros criterios.

El siguiente paso sería analizar la columna de “todos”. En ella se muestran el número de imágenes para las cuales cada método cumple los requisitos de las cuatro variables definidas.

En el mejor de los casos (*gradiente_ransac*) trece imágenes cumplen que la distancia del centro es menor a 20 pixels, que la diferencia del ángulo es menor al 10%, y que la diferencia de la excentricidad no se supera en un 9%.

Para conocer si es el mejor método, es decir, que no se necesita de ningún hibrido entre métodos para mejorar el resultado, se tiene que comprobar que el resto de imágenes que cumplen los datos para los diferentes métodos están contenidas en el conjunto de esas trece imágenes.

Se observó qué trece imágenes correspondían al cumplimiento de todos los requisitos, y luego cuales cumplían todos los requisitos en los demás métodos. A nada que una de las imágenes de cualquiera de los métodos no estuviera en el grupo de las trece, se consideró que ya podía existir un híbrido que mejorará los resultados.

Solo resultó necesario comparar con el primer método para ver que al menos una imagen no correspondía al grupo de trece, y en tal caso se creó un *hibrido*.

El *hibrido* que se utiliza para comprobar si se puede conseguir una mejoría de los resultados está formado por los tres métodos por los cuales se realiza la elección de puntos, y el algoritmo *ransac* para la elección de los parámetros de la elipse (fig.36).

El proceso se basa en realizar el cálculo de la mejor elipse para cada uno de los métodos. Al tener la mejor elipse, podemos conocer los puntos que se han utilizado para calcular dichos parámetros.

Una vez que se tienen los puntos con los que ha calculado la mejor elipse, se calcula la distancia algebraica entre los puntos y la mejor elipse, obteniendo de esta manera un error para cada método.

Los parámetros de la elipse final, lo formarán aquellos parámetros para los cuales el error haya sido mínimo, es decir, el método que menor distancia algebraica presente será el que aproxime la elipse con mayor exactitud.

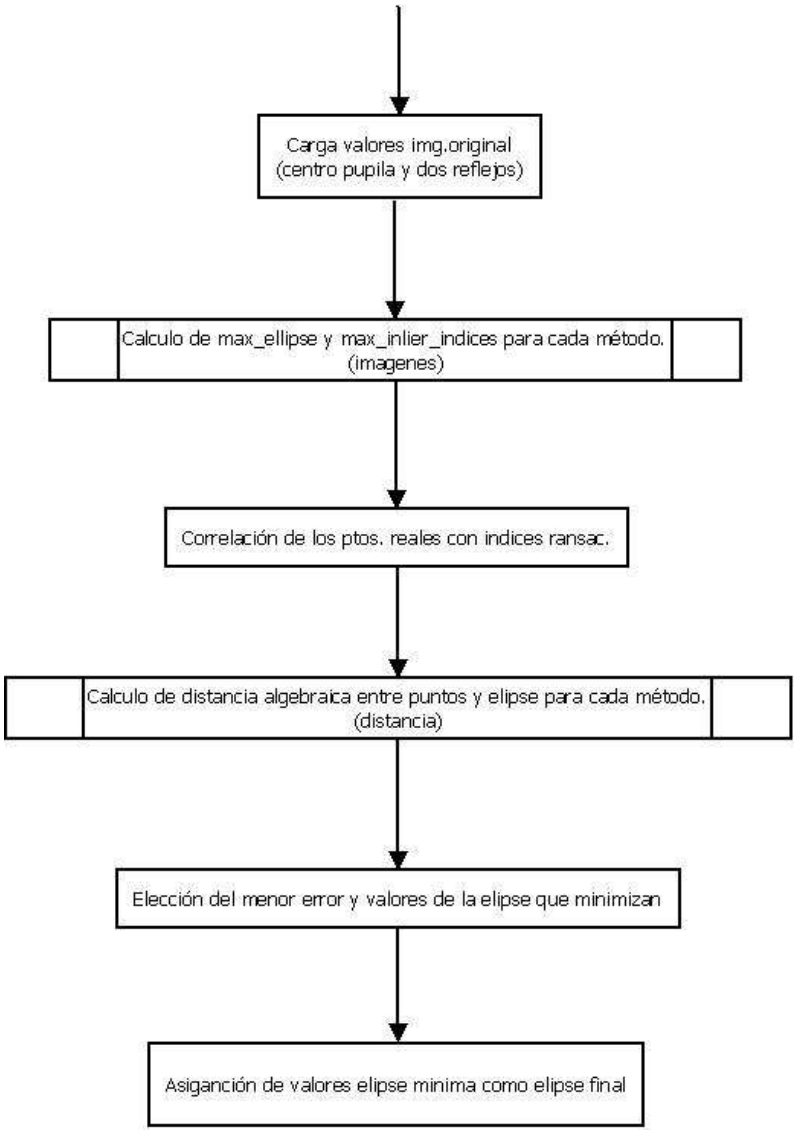


Fig.36. *Diagrama flujo hibrido*

Para comprobar si el algoritmo utilizado para el *hibrido* presenta una mejora, se debería observar las distancias algebraicas entre los mejores casos estudiados y el *hibrido*.

Primero se realizó el cálculo de todas las elipses y mediante la distancia algebraica que presentaban respecto a sus *inliers* se eligió el mejor método, en función del menor error. Tras tener la elipse, se calculaba una nueva distancia algebraica respecto a la *elipse_true*, la cual utilizaremos para comparar con el resto de métodos.

Con el cálculo de la distancia correcta, y con la ayuda de los datos que muestran valores significativos como la distancia media, la distancia máxima, la mínima y la desviación estándar (tabla 5) de los resultados calculados para los métodos se puede estudiar la evolución del *hibrido.*

	error_med	error_max	error_min	desviacion
gradiente_fit	123745,5653	20831981,39	418,8308479	1550170,478
gradiente_ransac	9358,755097	656139,5573	201,3228313	60931,40917
todos_fit	123753,1595	20833381,11	425,2827754	1550274,852
todos_ransac	13113,15837	684285,4127	200,8299002	78922,76973
umbral_fit	286696,6814	28788628,43	421,168555	2559845,741
umbral_ransac	71078,55248	850948,3883	208,7006863	186560,6355
hibrido	2145,428414	33122,98236	200,8299002	3086,153962

Tabla5. *Valores significativos distancias algebraicas*

Si observamos la fila del *hibrido* podemos observar que para cualquiera de los parámetros estudiados sus datos son los mínimos. Por lo que ello conlleva a que el menor error a la *elipse_true* se consigue mediante el hibrido, y la desviación estándar en dicho método también es la mínima, por lo que la diferencia de errores entre sus imágenes va a ser la que menos va a variar (fig.37).

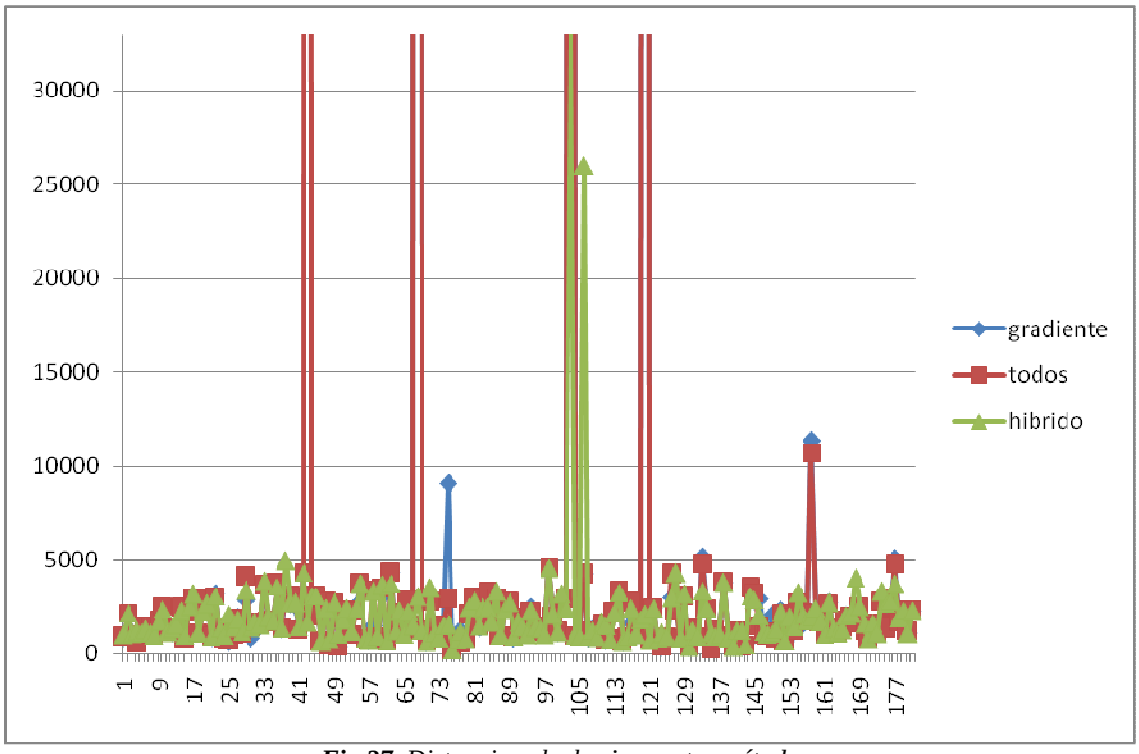


Fig.37. *Distancias algebraicas entre métodos.*

Se analizará si existe una existencia de mejora al utilizar el hibrido con respecto al modelo geométrico virtual 3D (tabla 6).

	centro_x	centro_y	phi	e	todos
gradiente_ransac	165	159	16	151	13
todos_ransac	166	159	13	148	11
umbral_ransac	137	139	7	150	6
hibrido	164	156	10	149	9

Tabla 6. *Nº imágenes que cumplen la sensibilidad*

Al observar el dato que nos indica la cantidad de imágenes que cumplen los cuatro parámetros que se estudian, el algoritmo no mejora, y de hecho, sería el que prácticamente peores resultados presentaría de todos los modelos (fig.38).

Estudiando cada parámetro por separado, el *hibrido* no presenta una mejora, por mínima que sea, en ninguno de los parámetros respecto a los mejores resultados obtenidos de cada algoritmo por separado.

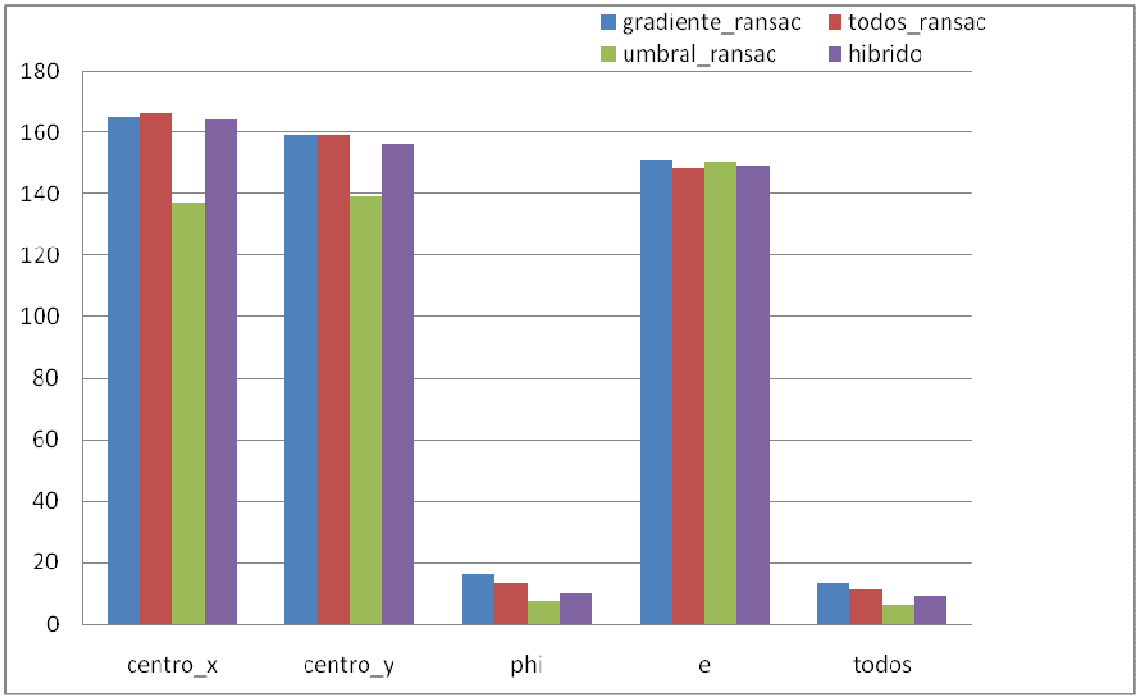


Fig.38. *Comparativa nº imágenes cumplen sensibilidad*

Bastante claro puede quedar la comparativa respecto al modelo geométrico, y que el mejor método a utilizar referente a la bondad que presenta respecto al modelo sería el *gradiente_ransac.*

Pero hay que observar que el cumplimiento de estos parámetros, no verifica que la aproximación a la elipse no sea mejor. Para ello hay que observar las diferencias por imágenes de los errores de cada método (fig. 35), en la que no se puede concluir nada claro.

Se presentan los datos de tal manera que no queda despejada la duda de cuál de los dos métodos muestra mejores resultados. Si analíticamente se realiza un estudio entre los

errores de ambos métodos, el 51,38% de las imágenes minimiza el error del *gradiente_ransac* frente a *todos_ransac*. Dicho valor, no es una cifra concluyente para poder elegir un buen método.

Se puede ultimar que para la mitad de las imágenes un método es mejor que otro, con la premisa de que si ahora realizásemos la prueba con otras imágenes diferentes, no sabríamos que resultados se obtendrían, por lo que no podemos decantarnos por ningún método.

Se podría cesar, diciendo que es mejor el *gradiente_ransac* al *todos_ransac* debido a que al comparar las imágenes con el modelo virtual, para el *gradiente_ransac* se obtienen más imágenes que superen los cuatro parámetros.

Pero, ¿qué queremos decir cuando nos referimos al mejor método? Si hacemos referencia de los resultados frente al modelo geométrico virtual, por número de aciertos, la solución sería *gradiente_ransac*. En cambio al referirnos a la elipse que más se aproxima a la elipse verdad, la respuesta no es tan fácil, ya que no existe un método claro que ayude a resolver la cuestión.

4.4 COMPARACIÓN INTERSUJETOS

Al realizar la comparación de los resultados expuestos anteriormente, se realizó la explicación del cálculo de la *elipse_true*.

Cabe recordar que dicha elipse estaba formada por los puntos del borde entre el iris y la esclera tras la marcación de dos sujetos distintos. Por lo que el siguiente análisis sería conocer, ¿Qué variabilidad existe entre las *elipse_true* de ambos sujetos? (fig.39).

Los resultados que se obtengan serán los umbrales, por lo que teóricamente será difícil superar los valores.

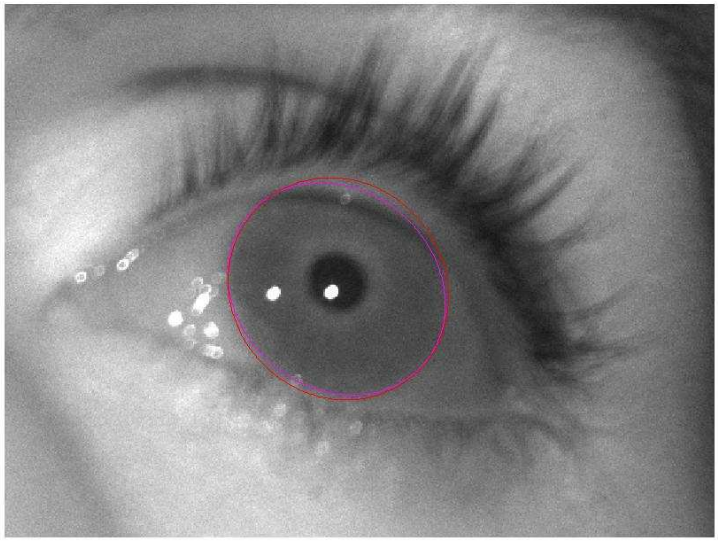


Fig.39. *Elipse_true de ambos sujetos*

Para ello se calculo una *elipse_true* para cada sujeto por separado, y luego se calculo la distancia algebraica entre ambas (fig.40).

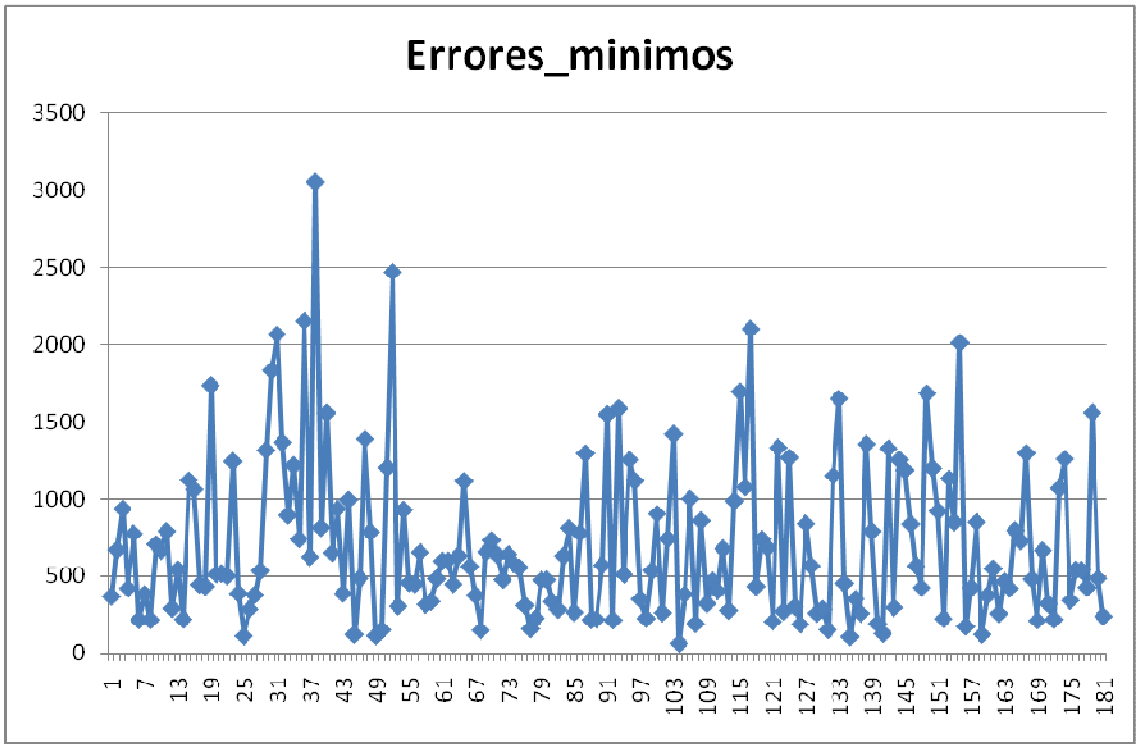


Fig.40. *Distancias algebraicas entre* ***elipse_true***

Para comprender mejor los resultados obtenidos respecto a la variación de distancias algebraicas es importante observar tanto la variación máxima que ocurre entre las dos *elipse_true* (fig. 41), como la mínima (fig.42).

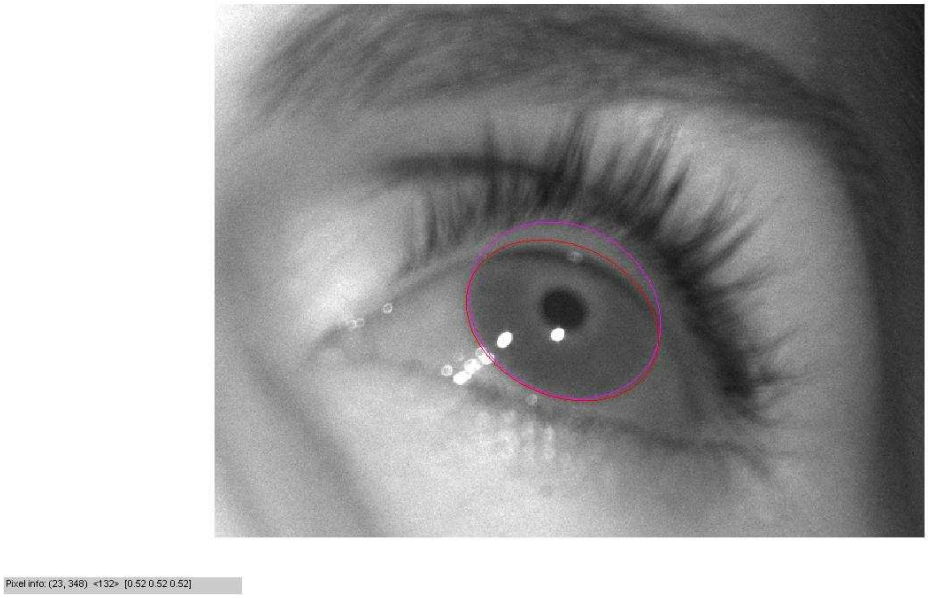


Fig.41. *Máxima variación* ***elipse_true***

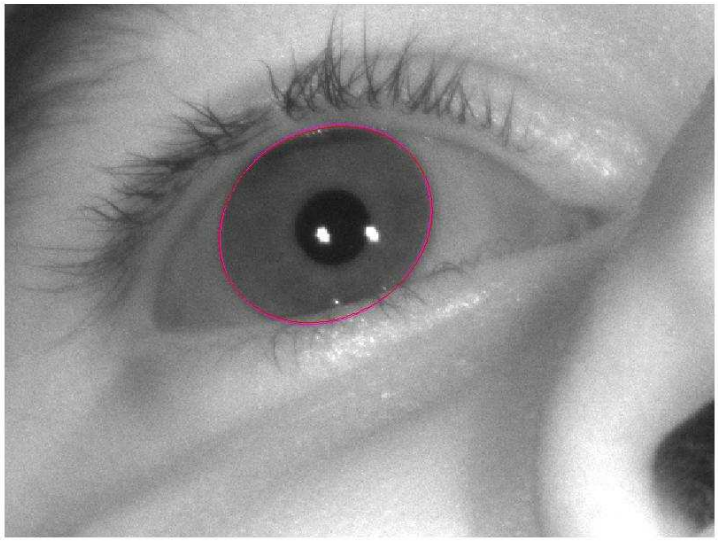


Fig.42. *Mínima variación* ***elipse_true***

En relación a las *elipse_true* calculadas, faltaría el estudio de la sensibilidad entre ellas respecto al modelo geométrico virtual.

Partiendo de que los puntos del sujeto uno se utilizarán como puntos finales de la *elipse_true* y los puntos marcados por el sujeto dos como puntos de la *elipse_calculada* se muestra la sensibilidad de los parámetros del modelo virtual (tabla 7).

	centro_x	centro_y	phi	e	todos
diferencias	181	181	26	132	26

Tabla 6. *Nº imágenes que cumplen la sensibilidad entre intersujetos*

5 CONCLUSIONES

Este proyecto tenía como finalidad la obtención del algoritmo para la detección aproximada de la elipse esclerocorneal en imágenes de Eye-tracking, para la cual sus resultados se han evaluado frente a dos sistemas.

Antes de comentar ambos sistemas, cabe destacar el uso referente al *canny*. Primariamente se utilizaron varios algoritmos para el cálculo de los bordes (operadores *sobel*, filtros de diferentes órdenes, derivadas) de la imagen, en los cuales se encontraban varias dificultades, y los bordes que resultaban no eran adecuados para la resolución del algoritmo.

Por lo que el uso del *canny*, resultó concluyente a la hora de la resolución, ya que los bordes que se obtienen, presentan dos picos claros referentes a las transiciones conocidas.

Uno de los sistemas de comparación ha sido el cálculo de la distancia algebraica, el cual permite conocer que método realiza mejor dicha aproximación. Por otro lado, se midió la fidelidad del algoritmo frente al modelo geométrico virtual 3D.

Para el primero de los sistemas no se puede concluir con que método se obtienen mejores resultados, pero sí que los mejores resultados van a estar abarcados por los métodos *gradiente_ransac* y *todos_ransac.*

A su vez se puede afirmar que el uso de cualquier método compuesto por *ransac*, obtendrá resultados más próximos a la *elipse_true.*

Relativo al sistema geométrico virtual 3D es más difícil analizar los resultados, pero discutiendo los métodos por separado el que mejor resultados presenta tanto en líneas generales, como en cualquiera de los parámetros estudiados es el *gradietne_ransac*.

Como se ha visto en los resultados, se realizó el uso de un *hibrido* el cual, deja algunos términos poco claros. Con dicho método se pretendía conseguir una mejora frente al cálculo aproximado de la *elipse_true,* y aunque esa mejora sea mínima, se ha conseguido.

Por otro lado la mejora frente a la bondad del sistema geométrico virtual 3D no es tan clara, por lo que no podemos concluir que el *hibrido* resulte una gran perfeccionamiento a todos los niveles.

6 LINEAS FUTURAS

Como base de dicho proyecto, se crea la realización de la captura de imágenes mediante el uso de reflectores infrarrojos, los cuales nos proporcionan dos reflejos en la imagen, que nos sirven como punto de partida con el centro de la pupila.

Como posteriores trabajos, se podría realizar el mismo proceso para el cálculo de la elipse esclerocorneal, pero sin la utilización de los reflectores infrarrojos, por lo que como punto de partida solo tendríamos el centro de la pupila.

En los métodos que se estudian para la elección de puntos, en el uso del gradiente la mayoría de la veces no es demasiado restrictivo, por lo que incluir una mayor restricción en el sería una posible mejora.

Realizar un estudio del sistema presentado con un mayor número de imágenes para las cuales la captura de ellas se podría efectuar mediante la calibración que realiza el Shakti, y poder realizar el estudio frente al modelo geométrico virtual 3D para diferentes puntos de la mirada de la pantalla.

Referente al *hibrido,* aunque en la aproximación a la *elipse_true* realice una mejora el tiempo de ejecución es bastante alto, por lo que *a priori* se debería decrementar este tiempo e incluso realizar algún ajuste que ayude a una exactitud mayor, para que así la bondad respecto al modelo geométrico virtual 3D también mejore.

BIBLIOGRAFÍA

[1] Andrew T Duchowsky, Eye Trucking methodology Theory and Practice, Springer

[2] D. Beymer,M. Flickner, Eye gaze tracking using an active stereo head, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, Wisconsin, 2003, pp. 451–458.

[3] Buswell, G. T. (1935). *How people look at pictures: a study of the psychology and perception in art*. Oxford, England: Univ. Chicago Press.

[4]C. Morimoto, A. Amir, and M. Flickner, "Detecting eye position and gaze from a single camera and 2 light sources," in *Proceedings. 16th International Conference on Pattern Recognition*, 2002, pp. 314–317.

[5] Delabarre, E. B. (1898). A method of recording eye-movements. *American Journal of Psychology, 9*(4), 572-574.

[6] Dodge, R., & Cline, T. S. (1901). The angle velocity of eye movements. *Psychological Review, 8*(2), 145-157.

[7] Dodge, R. (1906). Recent studies in the correlation of eye movement and visual perception. *Psychological Bulletin, 3*(3), 85-92.

[8] Dongheng Li, Derrick J. Parkhurst, STARBURST: A ROBUST ALGORTIHM FOR VIDEO-BASED EYE TRACKING

[9] Fender, D. H. (1964). Contact lens stability. *Biomedical and Scientific Instrumentation, 2*, 43-52.

[10] Huey, E. B. (1898). Preliminary Experiments in the Physiology and Psychology of Reading. *American Journal of Psychology, 9*(4), 575-886.

[11] Javal, E. (1879). Essai sur la Physiologie de la Lecture. *Annales D'Oculistique, 81*, 61-73.

[12]Jacob, R. J. K. (1991). The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look At is What You Get. *ACM Transactions on Information Systems, 9*(3), 152-169.

[13] R.J.K. Jacob, "Eye-movement-based human-computer interaction techniques: Towards non-command interfaces," 1993, pp. 151–190, Ablex Publishing corporation, Norwood, NJ.

[14] J. Merchant, R. Morrissette, J. Porterfield, Remote measurement of eye direction allowing subject motion over one cubic foot of space, IEEE Transactions on Biomedical Engineering 21 (4) (1974) 309–317.

[15] Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin, 124*(3), 372-422.

[16] Taylor, S. E. (1971). The dynamic activity of reading: A model of the process. In *EDL Research and Information Bulletin* (pp. 9). New York: McGraw-Hill.

[17] Yarbus, A. L. (1965). *Role of eye movements in the visual process*. Oxford, England: Nauka.

[18] http://en.wikipedia.org/wiki/Eye_tracking

[19] http://www.nosolousabilidad.com/articulos/eye-tracking.htm

[20] http://www.eyethink.org/publications_assets/EyeTrackingEBBE.pdf